

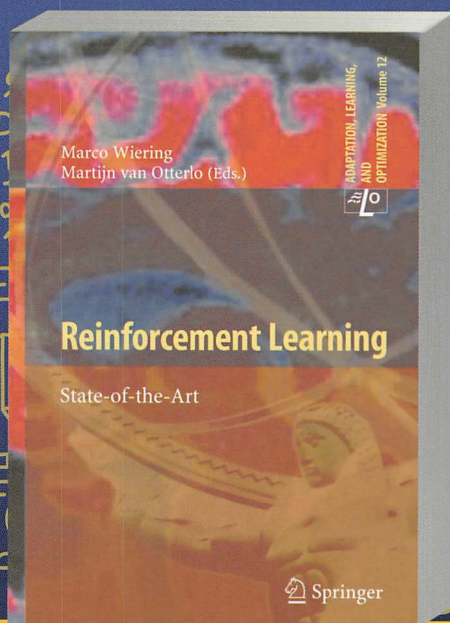
版权相关注意事项：

- 1、书籍版权归著者和出版社所有
- 2、本PDF来自于各个广泛的信息平台，经过整理而成
- 3、本PDF仅限用于非商业用途或者个人交流研究学习使用
- 4、本PDF获得者不得在互联网上以任何目的进行传播
- 5、如果觉得书籍内容很赞，请一定购买正版实体书，多多支持编写高质量的图书的作者和相应的出版社！当然，如果图书内容不堪入目，质量低下，你也可以选择狠狠滴撕裂本PDF
- 6、技术类书籍是拿来获取知识的，不是拿来收藏的，你得到了书籍不意味着你得到了知识，所以请不要得到书籍后就觉得沾沾自喜，要经常翻阅！！经常翻阅
- 7、请于下载PDF后24小时内研究使用并删掉本PDF

Reinforcement Learning State-of-the-Art

强化学习

[荷] 马可·威宁 (Marco Wiering) 编著
马丁·范·奥特罗 (Martijn van Otterlo)
赵地 刘莹 邓仰东 欧阳建权 苏统华 译



机械工业出版社
China Machine Press



图书在版编目 (CIP) 数据

强化学习 / (荷) 马可·威宁 (Marco Wiering), (荷) 马丁·范·奥特罗 (Martijn van Otterlo) 编著; 赵地等译. —北京: 机械工业出版社, 2018.6

(智能科学与技术丛书)

书名原文: Reinforcement Learning: State-of-the-Art

ISBN 978-7-111-60022-0

I. 强… II. ①马… ②马… ③赵… III. 机器学习 IV. TP181

中国版本图书馆 CIP 数据核字 (2018) 第 110203 号

本书版权登记号: 图字 01-2016-6249

Translation from the English language edition: *Reinforcement Learning: State-of-the-Art* edited by Marco Wiering and Martijn van Otterlo.

Copyright © Springer-Verlag Berlin Heidelberg 2012.

Springer is part of Springer Science+ Business Media.

All rights reserved.

本书中文简体字版由 Springer 授权机械工业出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

本书包括六大部分, 详细介绍了强化学习中各领域的基本理论和研究内容, 内容包括: MDP、动态规划、蒙特卡罗方法、批处理强化学习、TD 学习、Q 学习、策略迭代的最小二乘法、迁移学习、贝叶斯强化学习、一阶逻辑 MDP、层次式强化学习、演化计算、预测性定义状态表示、去中心化的部分可观察 MDP、博弈论和多学习器强化学习等内容, 并阐述强化学习与心理和神经科学、游戏领域、机器人领域的关系和应用, 有助于研究者了解强化学习领域, 发现新的研究方向。

本书适合作为高等院校机器学习、人工智能相关课程的参考书, 也可作为人工智能领域技术人员的参考用书。

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 唐晓琳

责任校对: 殷 虹

印 刷: 北京市兆成印刷有限责任公司

版 次: 2018 年 6 月第 1 版第 1 次印刷

开 本: 185mm×260mm 1/16

印 张: 30.5

书 号: ISBN 978-7-111-60022-0

定 价: 119.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzjsj@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东



强化学习在越来越多的实际问题中取得了突破性成果。基于强化学习的 AlphaGo 围棋程序连挫人类围棋冠军，赚足了眼球，而随后出现的新一代 AlphaGo Zero 则以 100:0 大败 AlphaGo。AlphaGo 是谷歌旗下 DeepMind 公司研发的人工智能下棋软件，主要由策略网络 (Policy Network)、快速走子 (Fast Rollout)、价值网络 (Value Network) 三个部分组成，并通过蒙特卡罗树搜索 (Monte Carlo Tree Search) 把三个部分有机连接，形成一个完整的系统。升级版的 AlphaGo Zero 最大限度地降低了人类棋谱的先验知识，完全通过强化学习的自我对弈提升棋力，青出于蓝而胜于蓝。现在强化学习的主攻热点转向了游戏以及机器人领域，强化学习在解决更多实际问题方面大有可为，同时也激发起强化学习研究领域的活力和热度。

强化学习是机器学习中与监督学习以及无监督学习平行的一种类型，它是 (自主) 智能体完成与外界环境交互任务的重要手段，通过最大化奖励函数的学习方法获取从环境状态到行为的映射函数。强化学习成为一个独立研究分支已有超过 50 年的历史，而 20 世纪 80 年代提出的马尔可夫决策过程 (Markov Decision Process, MDP) 构成了现代强化学习的基本描述框架。之后强化学习在理论、算法、应用上取得了长足的发展。对于真正想要在强化学习领域进行创新研究的学者而言，需要扎扎实实地研读强化学习方面的经典书籍和文献。

本书的编著者开篇就提出了目标：写一本值得向同学、同事及领域研究者推荐的讨论强化学习最新技术的好书。本书的特色鲜明，值得一读。第一是主题新颖。本书主要聚焦于发生在 2000 年到 2012 年间的最新发展。我们可以从第二~四部分看到发生在强化学习领域的最新动向和最新技术。撰写本书的作者以年轻学者为主，这也从一个侧面印证了本书的新颖度。第二是体例完整、涵盖的研究领域广泛。本书包含 19 章，其中第 1 章对强化学习的基本算法和框架做了全面的介绍，之后的 17 章对常规解决框架、构建性问题表示、概率建模手段以及经典应用领域进行详细评述，而最后一章则纵览全书进行讨论和发散。第三是组织精巧。内容从前到后具有一定的递增性，但又保持了各部分的相对独立性，方便读者根据兴趣选读相应篇章。最后，本书时刻围绕前沿性和开放性问题。作者在大胆发表自己的真知灼见的同时，不忘客观地审视当前的不足。这是本书不同于市面上很多书籍的重要特质。所以本书可以让你迅速跟上强化学习的发展现状。

本书的翻译工作由中国科学院计算机网络信息中心的赵地研究员发起并组建翻译团队。其中赵地研究员负责第 1、2 和 8 章的翻译工作，中国科学院大学的刘莹教授负责前言、第 3~7 章和第 12 章的翻译任务，清华大学的邓仰东教授承担第 9~11 章的翻译工作，湘潭大学的欧阳建权教授主持第 13~16 章的翻译，最后第 17~19 章的翻译由哈尔滨工业大学的苏统华教授完成。除了每章的负责人，还有很多研究生参与了部分翻译工作，特此向他们表示感谢。

本书几乎涵盖了经典强化学习的全部内容，甚至包括作为深度强化学习萌芽的重要成果 DFQ。但毕竟因时间问题未能及时顾及最近几年才发展出来的更多深度强化学习技术。我们



的翻译团队也期待能在未来再次合作，推出围绕深度强化学习的专著。

由于本书涉及的广度和深度较大，加上译者水平有限，译文中难免存在一些问题，真诚地希望读者朋友们批评指正。

最后要向机械工业出版社的朱劼编辑和唐晓琳编辑表示深深的谢意，她们在流程管理和文字编辑上提供的帮助对于本书的顺利出版至关重要。

2018 年 4 月



| 序 言 |

Reinforcement Learning: State-of-the-Art

强化学习是一门有 50 多年历史的学科，但是，由于受到马尔可夫决策过程理论的影响，其现代形式在 20 世纪 80 年代才逐渐兴起，并于 90 年代后期在教科书中建立起了完整的体系。在本书中，Martijn van Otterlo 和 Marco Wiering 这两位在该领域备受尊敬的、活跃的研究人员通过委托撰写，汇编出版了 21 世纪以来描述强化学习主要发展的一系列文章。这些文章都是综述而不是创新研究，每一篇都很权威地论述了强化学习的一个领域，包括神经和行为等方面的研究以及计算方面的考虑。对想要更进一步学习的学生和研究最新动态的科研人员来说，本书是一个宝贵的资源。

本人在这个领域已经工作了很长时间，这些文章的作者有两个突出的特点。第一，他们都很年轻。其中 16 篇文章的第一作者都是在过去 7 年内获得博士学位的（有些仍然是学生）。这无疑是一个非常好的信号，说明这个领域正在重生并十分具有活力。第二，三分之二的作者来自欧洲。部分原因是由于本书的编辑来自欧洲，这似乎也反映出强化学习研究的重心正在东移，从北美洲移到了欧洲。

Richard S. Sutton

2011 年 10 月



前言

Reinforcement Learning: State-of-the-Art

强化学习研究者们经常会被学生或同事问：“最近有没有一些强化学习方面的好书可以推荐给我？”

我们编写这本书的目的就是给这个问题提供一个答案。

一本关于强化学习的书

10年前上面的问题是很容易回答的，在那个时候，有两本时兴的权威书籍。一本是由 Rich Sutton 和 Andy Barto 在 1998 年编写的优秀的强化学习导论书籍^①。这本书从人工智能的角度出发，采用教科书式的写作风格，一直被广泛使用（截至目前引用了一万次）。另一本是 1996 年由 Dimitri Bertsekas 和 John Tsitsiklis 撰写的《神经动力学编程》(neuro-dynamic programming)^②。这本书从运筹学的角度出发，以精确的数学方法讲述了动态规划和强化学习，特别强调了求近似解的体系结构。其中 Sutton 和 Barto 总是最大化回报，谈及价值函数、回报，并偏向于使用 π 增加的字母表中的 $\{V, Q, S, A, T, R\}$ 部分；而 Bertsekas 和 Tsitsiklis 谈及代价函数 (cost-to-go-functions)，总是最小化成本，并且使用希腊符号 μ 增加的字母表中的 $\{J, G, I, U\}$ 部分。尽管它们有着表面（符号）差异、不同的写作风格和背景，可能这些书的读者也不同，但这两本书都试图对这个令人兴奋的新研究领域进行全面介绍，并成功地做到了这一点。当时运筹学和人工智能方法在行为优化方面的深入合并仍然在进行，这种交叉产生了丰硕的成果。最近，虽然已引入了 Q 学习和 TD 学习等强大的思想和算法，但仍有很多未知有待探索。

例如，算法和函数逼近器的组合的收敛问题出现了。包括算法收敛性、保证性能所需的样本数量以及强化学习技术在更大的智能体系结构中的适用性等许多理论和实验问题都没有得到解答。事实上，出现了许多新的问题并导致了越来越多的研究问题，这些都有待聪明的、年轻的博士生们来回答。尽管 Sutton 和 Barto、Bertsekas 和 Tsitsiklis 都很擅长介绍这个领域，并充分地描述了它的基本方法论和问题，但是，这个领域变得如此之大，需要新的教科书来记录所有新的研究进展。所以，这本书就是尝试填补这个空白的。

这是第一本介绍强化学习各主要子领域研究进展的书。但是，我们也提到其他一些有趣的介绍或描述各种强化学习主题的书籍。这些书包括 Leslie Kaelbling 于 1996 年编辑的合集^③和 Puterman 编著的马尔可夫决策过程手册的新版本^④。其他几本书^{⑤⑥}涉及近似动态规划的相

① Sutton and Barto (1998) Reinforcement Learning: An Introduction, MIT Press.

② Bertsekas and Tsitsiklis (1996) Neuro-Dynamic Programming, Athena Scientific.

③ L.P. Kaelbling (ed.) (1996) Recent Advances in Reinforcement Learning, Springer. Programming, Wiley.

④ M.L. Puterman (1994, 2005) Markov Decision Processes: Discrete Stochastic Dynamic

⑤ J. Si, A.G. Barto, W.B. Powell and D. Wunsch (eds.) (2004) Handbook of Learning and Approximate Dynamic Programming, IEEE Press.

⑥ W.B. Powell (2011) Approximate Dynamic Programming: Solving the Curses of Dimensionality, 2nd Edition, Wiley.



关概念。最近，又出现了一些关于马尔可夫决策过程^①、强化学习^②、函数逼近^③和强化学习的关系型知识表示^④的书籍。针对那些对强化学习课程感兴趣的人员，上述书只是强化学习相关著作的一部分。

强化学习：一个逐渐成熟的领域

在过去的 15 年中，强化学习领域发展迅猛。然而最近的书中并没有反映出这段时间的最新研究，而是更多地关注丰富的、坚实的理论研究，提升算法的适用性、向上扩展性、与（概率）人工智能的结合，以及大脑理论和一般的适应性系统的联系。现代强化学习的创始人之一 Richard Sutton^⑤，在 1999 年提出了强化学习发展的三个不同部分：过去、现在和未来。

过去的强化学习指的是 1985 年以前，在这个阶段，试错学习（trial-and-error learning）的思想得到了发展。这个时期强调使用积极探索的学习器（agent，也称智能体），并开发了利用标量回报信号来指定学习器目标的关键思想，称为回报假说。这些方法通常只学习策略，一般不能有效地处理延迟回报。

现在的强化学习指的是价值函数形成的时期。价值函数是强化学习的核心，几乎所有的方法都集中在价值函数的逼近上，以便计算（最优）策略。价值函数假说认为价值函数的逼近是智能化的主要目的。

目前，我们正处于强化学习的未来阶段。Sutton 对这个时期的方向做出了预测，并写道：“正如现在强化学习离开回报的最终目标向价值函数迈了一步，未来的强化学习可能会进一步把重点放在研究能够对价值函数进行估计的结构上……在心理学中，积极创造世界的表征的开发思维的方法称为建构主义。我预计在未来几十年中，强化学习将集中在建构主义上。”事实上，正如我们在本书中所看到的那样，这一领域的许多新进展都与能够实现价值函数逼近的新结构有关。此外，许多进展都是关于这些新结构的性能及收敛的性质、能力和保证的。贝叶斯框架、高效线性逼近、关系型知识表示以及分层和多学习器性质的分解都构成了当今强化学习方法中所采用的新结构。

目前强化学习是一个已经确立的研究领域，通常归于机器学习。然而，由于其专注于行为学习，它与心理学、运筹学、数学优化等其他领域有着许多联系。在人工智能领域，它与概率论和决策论规划有很大的重叠，因为它与规划社区（例如国际自动规划系统会议（ICAPS））有许多共同的目标。在最新的国际规划竞赛（IPC）中，源于强化学习文献的方法已经参赛，并且在概率规划问题和最近的“学习规划”（learning for planning）方面都有着非常好的表现。

强化学习的研究在人工智能的广泛领域中几乎随处可见，因为它既是行为优化的一般方法，也是一套计算工具。现在所有主要的人工智能期刊都发表关于强化学习的文章，并且已经持续很长时间了。强化学习的应用领域从机器人、电脑游戏到网络路由和自然语言对话

① O. Sigaud and O. Buffet (eds.) (2010) Markov Decision Processes in Artificial Intelligence, Wiley-ISTE.

② C. Szepesvari (2010) Algorithms for Reinforcement Learning, Morgan-Claypool.

③ L. Busoniu, R. Babuska, B. De Schutter and D. Ernst (2010) Reinforcement Learning and Dynamic Programming Using Function Approximators, CRC Press.

④ M. van Otterlo (2009) The Logic Of Adaptive Behavior, IOS Press.

⑤ R. S. Sutton (1999) Reinforcement Learning: Past, Present, Future – SEAL'98.



系统，强化学习论文也出现在跟这些主题相关的论坛上。大量的论文每年（或每两年）出现在人工智能领域的顶级会议上（如 IJCAI、ECAI 和 AAAI），还有许多统计机器学习领域的顶级会议上（如 UAI、ICML、ECML 和 NIPS）。此外，关于人工生命（Alife）、自适应行为（SAB）、机器人（ICRA、IROS、RSS）、神经网络和进化计算（如 IJCNN 和 ICANN）的会议也有强化学习的研究工作。最后但同样重要的一点是，在过去的 10 年中，所有主要的人工智能会议都出现了许多专业化的强化学习研讨会和教程。

尽管强化学习已经为其他许多领域做出了巨大贡献，并且强化学习的论文无处不在，但强化学习领域的现状使得它很自然地在强化学习方法的某个特定焦点上形成论坛。欧洲强化学习研讨会（EWRL）已经逐渐成为这样一个论坛，每隔一年就会有一次相当大的发展，2008 年在南锡举办并在 2011 年与 ECML 一起举办。此外，IEEE 自适应动态规划与强化学习（ADPRL）研讨会也成为研究人员展示和讨论其最新研究成果的一个会议。EWRL 和 ADPRL 一起表明，这一领域已经有了很大的进展，需要有自己的社区和事件。

在强化学习的实践方面以及更重要的是在基准、评估和比较方面也有了很大进展。除了规划比赛（例如 IPC）之外，一些强化学习比赛[Ⓔ]也已成功举办。参赛者不仅在几个经典领域进行竞赛（例如平衡杆），而且在电脑游戏“俄罗斯方块”和“超级马里奥”等新兴领域进行竞赛。这些比赛可以促进代码共享和重用，建立该领域的基准，并用于评估和比较具有挑战性的领域中的算法。另一个代码和解决方案重用的倡导者是 RL-Glue 框架[Ⓕ]，它提供了一个抽象的强化学习框架，用于在研究人员之间共享方法。RL-Glue 适用于大多数常用的编程语言，从而为实验提供了系统和语言独立的软件框架。比赛和 RL-Glue 促进了强化学习领域的成熟，使得可以应用更好的科学方法来测试、比较和重用强化学习方法。

本书的目的和目标读者

如前所述，我们试图让本书回答这个问题：“你会推荐什么样的书来学习目前的强化学习？”每个可能提出这个问题的人都是本书的潜在读者，这包括博士和硕士生、强化学习的研究人员，以及其他任何想了解强化学习领域的研究人员。书中关于当前强化学习主要研究领域的文献为研究人员提供了一个很好的起点去继续拓展该领域，把强化学习应用到新问题，并将主要的行为学习技术引入到他们自己的智能系统和机器人中。

当我们开始编著本书时，我们首先创建了一个长长的主题列表，并对它们进行了分组，最后选出了近 20 个比较大的强化学习子领域，这些子领域在过去 10 年里发布了许多新成果。这些子领域不仅包括比较成熟的子领域（如演进强化学习），还包括更新的子主题（如关系型知识表示方法、贝叶斯学习和规划框架）。此外，我们还专门用了一章来介绍分层方法，形成了第一个子领域——它是在前面提到的两本书之后出现的，因此当时没有讨论[Ⓖ]。

本书的理念是让所有的作者反映这个领域青春和活跃的本质。为此，我们主要选择并邀请了刚开始工作的年轻研究人员。他们中的许多人最近刚获得博士学位，这就确保了他们在自己的强化学习子领域是活跃的专家，并对这个子领域充满了想法和热情。而且，这也给了他们一个在更大的研究领域内推广其子领域研究成果的好机会。此外，我们还邀请了几位经

Ⓔ <http://www.rl-competition.org/>

Ⓕ glue.rl-community.org

Ⓖ 这不是说没有分层方法，但是当前分层技术的大部分都出现在 20 世纪 90 年代中期以后。



验丰富的研究人员，他们在强化学习的几个子领域取得了先进的研究成果。这一切使得关于这个主题的不同观点得到了很好的结合。正如我们所希望的那样，提交的内容初稿质量非常高。为了有一套确保高质量内容的完整程序，编辑组成员连同一批专家作为审稿人，对每章进行了至少三次审核。成书内容得到了进一步的改进，而且使书中包含了每个子领域的大量的参考文献。

本书的最终版本包含 19 章，其中第 1 章包含强化学习的基础知识、动态规划、马尔可夫决策过程和基础的算法（如 Q 学习和值迭代）。最后一章回顾了书中的内容，讨论了遗漏的东西，并指出了进一步研究的方向。另外，这一章还包含个人对这个领域的思考和预测。构成本书核心的 17 章中，每一章都是自成一体的，包含对强化学习子领域的介绍和概述。下面我们将会给出本书结构及各章的概要。本书共有 30 位作者，他们分别来自于不同的机构和不同的国家。

本书结构

这本书包含了 19 篇关于强化学习基础概念和各个子领域的综述，并分为四个主要的类别，我们接下来会对这些类别进行简要的说明。第 1 章由 Martijn van Otterlo 和 Marco Wiering 执笔，涵盖对基础概念与算法的介绍性材料。这一章讨论马尔可夫决策过程，及其对应的基于模型的和无模型的求解算法。这一章的目的是给读者提供一个快速了解强化学习方法主要构成的概述，同时该章也为其余各章提供了必要的背景知识。本书中的所有综述都建立在第 1 章的背景介绍的基础之上。本书的最后一章也是由 Marco Wiering 和 Martijn van Otterlo 执笔的，它回顾本书各章的内容，并列举了本书尚未讨论到的主题以及未来的研究方向。另外，通过汇总其他章部分作者的简要表述，列举了个人对强化学习领域的一些思考与预测。本书共有六个部分，其主体为第二～五部分，我们接下来将分别对它们进行简要的介绍。

第一部分（第 1 章）

这一部分对基础概念与算法进行了概述。

第二部分（第 2～6 章）

这一部分包含 5 章，介绍当前强化学习中使用的解决方案框架。其中所用到的大部分技术都能依据章节中定义的框架进行理解，尽管这些新方法侧重于以更加复杂的形式使用样本、世界模型等。

第 2 章由 Sascha Lange、Thomas Gabel 和 Martin Riedmiller 执笔，对价值函数逼近的上下文批处理强化学习方法进行了综述。这种方法能够利用高度优化的回归技术从海量的数据中学习到鲁棒的、精确的价值函数。第 3 章由 Lucian Buşoniu、Alessandro Lazaric、Mohammad Ghavamzadeh、Rémi Munos、Robert Babuška 和 Bart De Schutter 执笔，综合论述了强化学习在策略学习的鲁棒线性逼近技术方面的最新发展趋势。这些技术建立在一系列坚实的数学技巧之上，有这些数学基础的支撑，我们才可以建立学习速度、逼近精确度以及上下界的保证。第 4 章由 Todd Hester 和 Peter Stone 执笔，描述学习现实世界的模型的多种方法，以及这些模型如何加速强化学习。学习好的模型可以用来做更高效的值更新、做规划以及更有效的探索。世界模型代表着关于世界的一般知识，正因为如此，才有可能迁移到其

他相关的任务上。第5章由 Alessandro Lazaric 执笔, 详细介绍强化学习中的知识迁移。当遇到几个相关的任务时, 一旦学会了, 各种各样的事情可以在随后的任务中重用。例如, 策略可以重用, 但取决于两个相关任务的状态或动作空间是否不同, 需要应用其他方法。该章不仅考察了现有的方法, 而且试图把它们放在一个更普适的框架中。第6章由 Lihong Li 执笔, 对强化学习样本复杂度的技术和结果进行了综述。对于所有的算法, 了解需要多少个样本(与世界进行交互的例子)才能保证任务的最小性能是非常重要的。在过去的10年中, 鉴于利用严谨的数学方式研究这一重要方面出现了许多新的成果, 该章提供这些成果的概述。

第三部分(第7~10章)

这一部分包含4章, 其中表征及其构建和使用是重点内容。如前所述, 建设性的技术的一个主要方面是实现价值函数逼近的结构(或针对该问题的策略)。强化学习的几个主要新发展方向是寻找新的表征框架以用于在挑战性的新环境中学习行为。

第7章由 Hado van Hasselt 执笔, 描述了包含连续变量的问题表征的诸多技术。这在很长一段时间内都是强化学习的主要组成部分, 例如通过使用神经函数逼近器。然而, 该领域的一些新发展已经试图更严格地捕捉处理连续状态和动作的算法的性质, 或者已经将这样的技术应用于新的领域。我们特别感兴趣的是处理连续动作的新技术, 因为这些新技术能有效地使适用动作的数量无限大, 并需要复杂的技术来计算最优策略。第8章由 Martijn van Otterlo 执笔, 描述了10年前开始的强化学习的一个新的表征方向。该章涵盖了所有比状态和行为的命题(或属性-值)表达更加严格的表征。这些表征包括在逻辑编程和一阶逻辑中发现的模型。这样的表征可以用对象和关系来表示世界, 并且在一系列更广泛的领域中开辟了强化学习的可能性。这些表征开启了许多新的途径来泛化价值函数、策略以及世界模型, 并且需要逻辑机器学习和知识表示的方法来实现。第9章由 Bernhard Hengst 执笔, 综述一个具有代表性的研究方向, 而这里说的表征指的是某个任务的结构分解, 以及隐含的马尔可夫决策过程等方面。20世纪90年代末出现了许多分层方法, 从那以后产生了大量的技术。这些技术包括新的任务分解、价值函数和策略, 以及许多交互中自动学习的任务分解技术。第10章由 Shimon Whiteson 执笔, 综述良好的策略结构(和价值函数)的演进查询。这种演进算法对于迭代式的、增量式的强化学习方法一直是很好的替代, 而且两种方法都可以用来优化复杂行为。演进算法特别适合非马尔可夫问题和难以计算梯度的策略结构。除此之外, 该章还介绍应用在行为学习中的演进神经网络。

第四部分(第11~15章)

目前的人工智能已经变得越来越具有统计和概率的特点。概率图形模型领域的研究成果已经被广泛使用, 并且这些模型的结果(无论在理论上还是计算上)都有效地应用于许多子领域, 这与强化学习没有什么不同。有几个大的子领域在普遍使用概率模型, 如贝叶斯网络, 这种具有普遍性的表征和计算技术促使概率模型与其他相似的模型建立了丰富的联系。

第11章由 Nikos Vlassis、Mohammad Ghavamzadeh、Shie Mannor 和 Pascal Poupart 执笔, 综述用于强化学习的贝叶斯技术。在不确定情况下, 学习时序决策可以映射至贝叶斯空间, 其中, 交互痕迹提供样例(证据), 贝叶斯推理和学习可以通过一种严格的概率方式来寻找最优决策。第12章由 Matthijs Spaan 执笔, 综述了部分可观察的问题的表征和技术,

这些问题通常被映射到例如动态贝叶斯网络的概率框架中，并且需要概率推断来推测潜在的隐藏（未观察的）状态。该章同时综述了基于模型的和无模型的方法。鉴于 POMDP 通常根据某种形式的历史（或记忆）的置信状态来建模，由 David Wingate 执笔的第 13 章综述了最近的一类侧重于未来的新方法。这些技术维护一个置信状态，用于根据对未来事件的概率预测做出行为选择。该章介绍了几种技术，其中这些预测用简洁的方式表示，并根据经验进行更新。到目前为止，大多数方法都集中在预测（或评估）问题上，而控制方面的论述很少。第 14 章由 Ann Nowé、Peter Vrancx 和 Yann-Michaël De Hauwere 执笔，转移到一系列更通用的问题——多个学习器的学习和交互。该章综述博弈论和多学习器方法，介绍用于优化多学习器的技术。第 15 章由 Frans Oliehoek 执笔，综述由多个学习器组成的基于模型的技术，这些学习器合作解决由 POMDP 分解的大任务。这种模型出现在如何优化不同地点传感器共同提供世界重要信息方式的问题中。该章主要介绍 POMDP 方法和多学习器的情况。

第五部分（第 16 ~ 18 章）

正如我们在前言开始所说的那样，强化学习是一种在人工智能的许多其他领域用来优化行为的方法。因此，除了本书前面部分介绍的许多先进的算法之外，我们还将包括强化学习取得成功的一些应用领域。这一部分的特色是介绍机器人和游戏，还有一章介绍了将强化学习与认知神经科学联系起来的研究方向。

第 16 章由 Ashvin Shah 执笔，综述了强化学习方法与认知和神经科学之间的关系。最初，许多强化学习的技术来源于心理学方面的见解，例如 Skinner、Thorndike 和 Watson 等的见解，还有心理学和强化学习之间的交叉领域。最近，由于脑科学理论的进步，尤其是因为测试和测量脑部活动（功能核磁共振成像、脑电图等）已经变得更成熟，很多研究试图解释有关大脑的强化学习方面的研究成果、学习技术，即哪些算法确实出现在大脑中，或者受大脑内部运作启发而提出新的算法。第 17 章由 István Szita 执笔，综述游戏中使用的强化学习。在这里“游戏”比前面关于博弈论的章节中的更通用。实际上，该章中的游戏相当于棋盘游戏，如西洋双陆棋和跳棋，还包括角色扮演和实时策略游戏等电脑游戏。游戏通常是一个令人兴奋的强化学习算法的测试平台（例如参见上述强化学习竞赛中的“俄罗斯方块”和“马里奥”），除了举出很多例子外，该章还试图勾勒出强化学习在游戏中的应用。第 18 章由 Jens Kober 和 Jan Peters 执笔，严谨地描述了强化学习在机器人中的应用。由于机器人技术在真实的物理世界中工作，产生了许多有挑战性的问题。大量的噪声数据、真实机器人的训练和测试缓慢、模拟器与现实世界之间的实际差距以及扩展到高维空间等，这些都是这里所讨论的具有挑战性的问题。机器人技术是一个令人兴奋的领域，因为将人类置于其中的可能性可以为仿生学创造额外的机会，从示范中学习，并让人类作为机器人的教师。

第六部分（第 19 章）

这一部分对全书进行总结，并展望了强化学习的未来。

致谢

编写这样的一本书不是一件一蹴而就的事情。许多人为此付出了非常多的努力。首先，我们要感谢所有的作者们，他们付出专业知识、时间以及创造力对各自的子领域进行了精彩

的论述。撰写综述通常需要付出格外多的努力，因为这需要你非常了解某个主题，而且需要你将所有相关的工作放在更加通用的框架中。作为编辑，我们非常高兴地看到作者们圆满完成了这个困难但却非常有用的任务。

我们想感谢的第二群人是审稿人，他们为我们提供了非常透彻且非常具有建设性的评论，使得这本书更加完美。我们感谢那些同意在书中写下自己名字的审稿人，非常感谢你们的帮助：Andrea Bonarini, Prasad Tadepalli, Sarah Ostentoski, Rich Sutton, Daniel Kudenko, Jesse Hoey, Christopher Amato, Damien Ernst, Remi Munos, Johannes Fuernkrantz, Juergen Schmidhuber, Thomas Rückstieß, Joelle Pineau, Dimitri Bertsekas, John Asmuth, Lisa Torrey, Yael Niv, Te Thamrongrattanarit, Michael Littman 和 Csaba Szepesvari。

非常感谢 Rich Sutton 为本书写下序言。我们都认为他是强化学习领域的领军人物，而且，我们都钦佩他在所有方面为这个领域所做出的巨大贡献。他在强化学习刚刚兴起的时候就开始研究，并且不断地提出新颖的、有创造性的方法让学习器去学习。感谢 Rich 先生！

如果能够把编辑这样一本书纳入日常科学生活中，那会更加方便。在这方面，Martijn 要感谢比利时鲁汶大学和荷兰奈梅亨大学的支持。Marco 也非常感谢荷兰格罗宁根大学提供同样的支持。

最后，我们要感谢读者选择了这本书并开始阅读。我们希望这本书能够为你提供帮助，并希望你即将开始的工作会被纳入随后的强化学习书籍中。

Marco Wiering, 荷兰格罗宁根大学人工智能系

Martijn van Otterlo, 荷兰奈梅亨大学

2011 年 11 月

Robert Babuška

荷兰代尔夫特理工大学代尔夫特系统与控制中心

e-mail: r.babuska@tudelft.nl

Lucian Buşoniu

法国洛林大学自动控制研究中心 (CRAN)

e-mail: lucian@busoniu.net

Thomas Gabel

德国弗莱堡大学工程学院

e-mail: tgabel@informatik.uni-freiburg.de

Mohammad Ghavamzadeh

法国 INRIA Lille-Nord SequeL 团队

e-mail: mohammad.ghavamzadeh@inria.fr

Hado van Hasselt

荷兰数学和计算机中心 (Centrum Wiskunde en Informatica, CWI)

e-mail: H.van.Hasselt@cwi.nl

Yann-Michaël De Hauwere

比利时布鲁塞尔自由大学

e-mail: ydehauwe@vub.ac.be

Bernhard Hengst

澳大利亚新南威尔士大学计算机科学与工程学院

e-mail: bernhardh@cse.unsw.edu.au

Todd Hester

得克萨斯大学奥斯汀分校计算机科学系

e-mail: todd@cs.utexas.edu

Jens Kober

德国达姆施塔特工业大学智能自治系统研究所,

德国马克斯普朗克智能系统研究所机器人学习实验室

e-mail: jens.kober@tuebingen.mpg.de

Sascha Lange

德国弗莱堡大学工程学院

e-mail: slange@informatik.uni-freiburg.de

Alessandro Lazaric

法国 INRIA Lille-Nord SequeL 团队

e-mail: alessandro.lazaric@inria.fr

Lihong Li

美国雅虎研究院

e-mail: lihong@yahoo-inc.com

Shie Mannor

以色列理工学院

e-mail: shie@ee.technion.ac.il

Rémi Munos

法国 INRIA Lille-Nord SequeL 团队

e-mail: remi.munos@inria.fr

Frans Oliehoek

麻省理工学院计算机科学与人工智能实验室 (CSAIL)

e-mail: fao@csail.mit.edu

Ann Nowé

比利时布鲁塞尔自由大学

e-mail: anowe@vub.ac.be

Martijn van Otterlo

荷兰奈梅亨大学

e-mail: m.vanotterlo@donders.ru.nl

Jan Peters

德国达姆施塔特工业大学智能自治系统研究所,

德国马克斯普朗克智能系统研究所机器人学习实验室

e-mail: jan.peters@tuebingen.mpg.de

Pascal Poupart

加拿大滑铁卢大学

e-mail: ppoupart@cs.uwaterloo.ca

Martin Riedmiller

德国弗莱堡大学工程学院

e-mail: riedmiller@informatik.uni-freiburg.de

Bart De Schutter

荷兰代尔夫特理工大学代尔夫特系统与控制中心

e-mail: b.deschutter@tudelft.nl

Ashvin Shah

英国谢菲尔德大学心理学系

e-mail: ashvin@gmail.com

Matthijs Spaan

葡萄牙里斯本技术大学系统与机器人研究所

e-mail: mtjspaan@isr.ist.utl.pt

Peter Stone

得克萨斯大学奥斯汀分校计算机科学系

e-mail: pstone@cs.utexas.edu

István Szita

加拿大阿尔伯塔大学

e-mail: szityu@gmail.com

Nikos Vlassis

卢森堡大学系统生物医学研究中心,

卢森堡 OneTree 公司

e-mail: nikos.vlassis@uni.lu,nikos@onetreesol.com

Peter Vrancx

比利时布鲁塞尔自由大学

e-mail: pvrancx@vub.ac.be

Shimon Whiteson

荷兰阿姆斯特丹大学信息学研究所

e-mail: s.a.whiteson@uva.nl

Marco Wiering

荷兰格罗宁根大学人工智能系

e-mail: m.a.wiering@rug.nl

David Wingate

麻省理工学院

e-mail: wingated@mit.edu

目 录

Reinforcement Learning: State-of-the-Art

译者序
序言
前言
作者清单

第一部分 绪论

第 1 章 强化学习和马尔可夫决策

过程	2
1.1 简介	2
1.2 时序决策	3
1.2.1 接近时序决策	4
1.2.2 在线学习与离线学习	4
1.2.3 贡献分配	5
1.2.4 探索-运用的平衡	5
1.2.5 反馈、目标和性能	5
1.2.6 表达	6
1.3 正式的框架	6
1.3.1 马尔可夫决策过程	7
1.3.2 策略	9
1.3.3 最优准则和减量	9
1.4 价值函数和贝尔曼方程	10
1.5 求解马尔可夫决策过程	12
1.6 动态规划：基于模型的解决	
方案	13
1.6.1 基本的动态规划算法	13
1.6.2 高效的动态规划算法	17
1.7 强化学习：无模型的解决方案	19
1.7.1 时序差分学习	20
1.7.2 蒙特卡罗方法	23
1.7.3 高效的探索和价值更新	24
1.8 总结	27
参考文献	27

第二部分 高效的解决方案框架

第 2 章 批处理强化学习

2.1 简介	32
2.2 批处理强化学习问题	33
2.2.1 批处理学习问题	33
2.2.2 增长批处理学习问题	34
2.3 批处理强化学习算法的基础	34
2.4 批处理强化学习算法	37
2.4.1 基于核的近似动态规划	37
2.4.2 拟合 Q 迭代	39
2.4.3 基于最小二乘的策略迭代	40
2.4.4 识别批处理算法	41
2.5 批处理强化学习理论	42
2.6 批处理强化学习的实现	43
2.6.1 神经拟合 Q 迭代	44
2.6.2 控制应用中的神经拟合 Q 迭代	
算法	45
2.6.3 面向多学习器的批处理	
强化学习	46
2.6.4 深度拟合 Q 迭代	48
2.6.5 应用/发展趋势	49
2.7 总结	50
参考文献	50

第 3 章 策略迭代的最小二乘法

3.1 简介	53
3.2 预备知识：经典策略迭代算法	54
3.3 近似策略评估的最小二乘法	55
3.3.1 主要原则和分类	55
3.3.2 线性情况下和矩阵形式的	
方程	57
3.3.3 无模型算法的实现	60
3.3.4 参考文献	62

3.4 策略迭代的在线最小二乘法	63
3.5 例子: car-on-the-hill	64
3.6 性能保障	66
3.6.1 渐近收敛性和保证	66
3.6.2 有限样本的保证	68
3.7 延伸阅读	73
参考文献	74

第4章 学习和使用模型 78

4.1 简介	78
4.2 什么是模型	79
4.3 规划	80
4.4 联合模型和规划	82
4.5 样本复杂度	84
4.6 分解域	86
4.7 探索	88
4.8 连续域	91
4.9 实证比较	93
4.10 扩展	95
4.11 总结	96
参考文献	97

第5章 强化学习中的迁移: 框架和概观 101

5.1 简介	101
5.2 强化学习迁移的框架和分类	102
5.2.1 迁移框架	102
5.2.2 分类	104
5.3 固定状态 – 动作空间中从源到目标迁移的方法	108
5.3.1 问题形式化	108
5.3.2 表示迁移	109
5.3.3 参数迁移	110
5.4 固定状态 – 动作空间中跨多任务迁移的方法	111
5.4.1 问题形式化	111
5.4.2 实例迁移	111
5.4.3 表示迁移	112
5.4.4 参数迁移	113
5.5 不同状态 – 动作空间中从源到	

目标任务迁移的方法	114
5.5.1 问题形式化	114
5.5.2 实例迁移	115
5.5.3 表示迁移	115
5.5.4 参数迁移	116
5.6 总结和开放性问题	116
参考文献	117

第6章 探索的样本复杂度边界 122

6.1 简介	122
6.2 预备知识	123
6.3 形式化探索效率	124
6.3.1 探索的样本复杂度和 PAC-MDP	124
6.3.2 遗憾最小化	125
6.3.3 平均损失	127
6.3.4 贝叶斯框架	127
6.4 通用 PAC-MDP 定理	128
6.5 基于模型的方法	130
6.5.1 Rmax	130
6.5.2 Rmax 的泛化	132
6.6 无模型方法	138
6.7 总结	141
参考文献	141

第三部分 建设性的表征方向

第7章 连续状态和动作空间中的强化学习 146

7.1 简介	146
7.1.1 连续域中的马尔可夫决策过程	147
7.1.2 求解连续 MDP 的方法	148
7.2 函数逼近	149
7.2.1 线性函数逼近	150
7.2.2 非线性函数逼近	153
7.2.3 更新参数	154
7.3 近似强化学习	157
7.3.1 数值逼近	157
7.3.2 策略逼近	162

7.4 双极车杆实验	168	9.3.2 HAMQ 学习	222
7.5 总结	171	9.3.3 MAXQ	223
参考文献	171	9.4 学习结构	226
第 8 章 综述：求解一阶逻辑		9.5 相关工作和当前研究	228
马尔可夫决策过程	179	9.6 总结	230
8.1 关系世界中的顺序决策简介	179	参考文献	230
8.1.1 马尔可夫决策过程：代表性 和可扩展性	180	第 10 章 针对强化学习的演化	
8.1.2 简短的历史和与其他领域的 联系	181	计算	235
8.2 用面向对象和关系扩展马尔可夫 决策过程	183	10.1 简介	235
8.2.1 关系表示与逻辑归纳	183	10.2 神经演化	237
8.2.2 关系型马尔可夫决策过程	184	10.3 TWEANN	239
8.2.3 抽象问题和求解	184	10.3.1 挑战	239
8.3 基于模型的解决方案	186	10.3.2 NEAT	240
8.3.1 贝尔曼备份的结构	186	10.4 混合方法	241
8.3.2 确切的基于模型的算法	187	10.4.1 演化函数近似	242
8.3.3 基于近似模型的算法	190	10.4.2 XCS	243
8.4 无模型的解决方案	192	10.5 协同演化	245
8.4.1 固定泛化的价值函数学习	192	10.5.1 合作式协同演化	245
8.4.2 带自适应泛化的价值函数	193	10.5.2 竞争式协同演化	246
8.4.3 基于策略的求解技巧	196	10.6 生成和发展系统	247
8.5 模型、层级、偏置	198	10.7 在线方法	249
8.6 现在的发展	201	10.7.1 基于模型的技术	249
8.7 总结和展望	203	10.7.2 在线演化计算	250
参考文献	204	10.8 总结	251
第 9 章 层次式技术	213	参考文献	251
9.1 简介	213	第四部分 概率模型	
9.2 背景	215	第 11 章 贝叶斯强化学习	260
9.2.1 抽象动作	215	11.1 简介	260
9.2.2 半马尔可夫决策问题	216	11.2 无模型贝叶斯强化学习	261
9.2.3 结构	217	11.2.1 基于价值函数的算法	261
9.2.4 状态抽象	218	11.2.2 策略梯度算法	264
9.2.5 价值函数分解	219	11.2.3 演员-评论家算法	266
9.2.6 优化	220	11.3 基于模型的贝叶斯强化学习	268
9.3 层次式强化学习技术	220	11.3.1 由 POMDP 表述的贝叶斯 强化学习	268
9.3.1 选项	221	11.3.2 通过动态规划的贝叶斯 强化学习	269

11.3.3 近似在线算法	271
11.3.4 贝叶斯多任务强化学习	272
11.3.5 集成先验知识	273
11.4 有限样本分析和复杂度问题	274
11.5 总结和讨论	275
参考文献	275

第 12 章 部分可观察的马尔可夫决策过程

12.1 简介	279
12.2 部分可观察环境中的决策	280
12.2.1 POMDP 模型	280
12.2.2 连续和结构化的表达	281
12.2.3 优化决策记忆	282
12.2.4 策略和价值函数	284
12.3 基于模型的技术	285
12.3.1 基于 MDP 的启发式解决方案	285
12.3.2 POMDP 的值迭代	286
12.3.3 确切的值迭代	288
12.3.4 基于点的值迭代方法	290
12.3.5 其他近似求解方法	291
12.4 无先验模型的决策	292
12.4.1 无记忆技术	292
12.4.2 学习内部记忆	292
12.5 近期研究趋势	294
参考文献	295

第 13 章 预测性定义状态表示

13.1 简介	300
13.1.1 状态是什么	301
13.1.2 哪一个状态表示	301
13.1.3 为什么使用预测性定义模型	302
13.2 PSR	303
13.2.1 历史及测试	303
13.2.2 测试的预测	304
13.2.3 系统动态向量	304
13.2.4 系统动态矩阵	305
13.2.5 充分的数据集	305

13.2.6 状态	306
13.2.7 更新状态	306
13.2.8 线性 PSR	307
13.2.9 线性 PSR 与 POMDP 的关联	307
13.2.10 线性 PSR 的理论结果	308
13.3 PSR 模型学习	308
13.3.1 发现问题	308
13.3.2 学习问题	309
13.3.3 估计系统动态矩阵	309
13.4 规划与 PSR	309
13.5 PSR 的扩展	310
13.6 其他具有预测性定义状态的模型	311
13.6.1 可观测量子模型	311
13.6.2 预测线性高斯模型	312
13.6.3 时序差分网络	312
13.6.4 分集自动机	312
13.6.5 指数族 PSR	313
13.6.6 转换 PSR	313
13.7 总结	313
参考文献	314

第 14 章 博弈论和多学习器强化学习

14.1 简介	317
14.2 重复博弈	319
14.2.1 博弈论	319
14.2.2 重复博弈中的强化学习	322
14.3 顺序博弈	325
14.3.1 马尔可夫博弈	326
14.3.2 马尔可夫博弈中的强化学习	327
14.4 在多学习器系统中的稀疏交互	330
14.4.1 多等级学习	330
14.4.2 协调学习与稀疏交互	331
14.5 延伸阅读	334
参考文献	334

第 15 章 去中心化的部分可观察

马尔可夫决策过程	338
15.1 简介	338
15.2 Dec-POMDP 框架	339
15.3 历史状态与策略	340
15.3.1 历史状态	341
15.3.2 策略	341
15.3.3 策略的结构	342
15.3.4 联合策略的质量	343
15.4 有限域的 Dec-POMDP 的 解决方案	344
15.4.1 穷举搜索和 Dec-POMDP 复杂性	344
15.4.2 交替最大化	344
15.4.3 Dec-POMDP 的最优价值 函数	345
15.4.4 前推法: 启发式搜索	348
15.4.5 后推法: 动态规划	350
15.4.6 其他有限域的方法	353
15.5 延伸阅读	353
15.5.1 一般化和特殊问题	353
15.5.2 有限 Dec-POMDP	354
15.5.3 强化学习	355
15.5.4 通信	356
参考文献	356

第五部分 其他应用领域

第 16 章 强化学习与心理和神经科学
之间的关系

之间的关 系	364
16.1 简介	364
16.2 经典 (巴甫洛夫) 条件反射	365
16.2.1 行为	365
16.2.2 理论	366
16.2.3 小结和其他注意事项	367
16.3 操作性 (工具性) 条件反射	368
16.3.1 动作	368
16.3.2 理论	369
16.3.3 基于模型的控制与无模型的 控制	370

16.3.4 小结和其他注意事项	371
16.4 多巴胺	371
16.4.1 多巴胺作为奖励预测误差	372
16.4.2 多巴胺的强化信号的作用	372
16.4.3 小结和其他注意事项	373
16.5 基底神经节	373
16.5.1 基底神经节概述	374
16.5.2 纹状体的神经活动	374
16.5.3 皮质基神经节丘脑循环	375
16.5.4 小结和其他注意事项	377
16.6 总结	378
参考文献	378

第 17 章 游戏领域的强化学习

17.1 简介	387
17.1.1 目标和结构	387
17.1.2 范围	388
17.2 游戏展示厅	388
17.2.1 西洋双陆棋	389
17.2.2 国际象棋	391
17.2.3 围棋	394
17.2.4 俄罗斯方块	398
17.2.5 即时战略游戏	400
17.3 强化学习应用到游戏的挑战	402
17.3.1 表示的设计	402
17.3.2 探索	404
17.3.3 训练数据的来源	405
17.3.4 处理缺失的信息	406
17.3.5 对手建模	407
17.4 在游戏中使用强化学习	407
17.4.1 最具娱乐性的对手	407
17.4.2 开发期间的学习	408
17.5 总结	409
参考文献	410

第 18 章 机器人领域的强化学习

综 述	415
18.1 简介	415
18.2 机器人强化学习中的挑战	416
18.2.1 维度灾难	417

18.2.2	真实场景样本灾难	418
18.2.3	真实场景交互灾难	418
18.2.4	模型错误灾难	418
18.2.5	目标规范灾难	419
18.3	机器人强化学习基础	419
18.3.1	价值函数方法	420
18.3.2	策略搜索	421
18.4	表示法带来的可行性	422
18.4.1	智能状态-动作离散化	423
18.4.2	函数近似	423
18.4.3	预构建策略	424
18.5	先验知识带来的可行性	425
18.5.1	示范中的先验知识	425
18.5.2	任务结构中的先验知识	426
18.5.3	先验知识指导探索	427
18.6	仿真模拟带来的可行性	427
18.6.1	模型的作用	427
18.6.2	智力预演	428
18.6.3	从仿真直接迁移到真实 机器人	429
18.7	一个学习样例: 杯中球任务	429
18.7.1	实验设置: 任务和奖励	429
18.7.2	适当的策略表示	430
18.7.3	生成教师的示范	430

18.7.4	使用策略搜索进行强化 学习	430
18.7.5	机器人强化学习中使用 仿真模拟	431
18.7.6	价值函数方法的替代方案	431
18.8	总结	432
	参考文献	432

第六部分 结束语

第 19 章 总结、未来方向和展望 440

19.1	回顾	440
19.1.1	本书覆盖内容	440
19.1.2	哪些主题没有被包含	441
19.2	展望未来	445
19.2.1	目前未知的内容	445
19.2.2	看起来不可能的强化学习 应用	446
19.2.3	有趣的方向	447
19.2.4	专家对未来发展的看法	448
	参考文献	449

缩写词	453
-----	-----

索引	455
----	-----

绪 论

强化学习和马尔可夫决策过程

Martijn van Otterlo, Marco Wiering

摘要

在监督学习 (supervised learning) 和无监督学习 (unsupervised learning) 之间, 强化学习的模式可以处理时序决策问题中的学习, 而这些决策问题只能得到有限的反馈。本章介绍了马尔可夫决策过程和用于计算最优行为的两类算法的直观认识及概念: 强化学习和动态规划。首先, 定义了马尔可夫决策过程的形式化框架, 随后定义了价值函数 (value function) 和策略。

本章根据最优性的各种定义以及时序决策学习的目标, 主要介绍了用于学习最优行为的算法的基本类别。此外, 本章综述了基本算法的有效扩展, 主要是在环境中的反馈方式的不同来加快学习, 并在方式上, 他们集中在有关部分的问题。对于基于模型的设置和模型无关的设置, 这些有效扩展已证明在扩展到更大的问题时很有用。

1.1 简介

马尔可夫决策过程 (Markov Decision Processes, MDP) [Puterman, 1994] 是决策理论规划 (Decision-Theoretic Planning, DTP) [Boutilier et al, 1999; Boutilier, 1999]、强化学习 (Reinforcement Learning, RL) [Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998; Kaelbling et al, 1996] 及随机域中的其他学习问题的一种直观和基本的构造模型。在这个模型中, 环境通过一组状态和动作进行建模, 可以用来执行以控制系统的状态。通过这种方式来控制系统的目的是最大化一个模型的性能标准。很多问题 (例如随机规划问题、学习机器人控制和玩游戏的问题) 已成功地通过马尔可夫决策过程建模。事实上, 马尔可夫决策过程已经成为时序决策 (Sequential Decision Making, SDM) 事实上的标准方法。

决策理论规划 [Boutilier et al, 1999] (例如, 使用决策理论的概念来表示不确定性和规划的质量) 是人工智能 (AI) 规划范式的一个重要的扩展。这个扩展添加了处理动作效应的不确定性的能力和处理没有清晰定义的目标函数的能力。此外, 因为考虑了资源消耗和质量变化时的不确定性需求, 例如质量的变化可能发生在实时决策的场景中, 决策理论规划显著地增加了模型的维度。人工智能规划、运筹学 (operations research) [Winston, 1991] 和控制理论 (control theory) [Bertsekas, 1995] 之间有着许多联系。例如, 时序决策领域的许多工作可以看成马尔可夫决策过程的实例。人工智能规划中规划的概念 (即, 从起始状态 (start state) 到目标状态 (goal state) 的一系列动作) 已经扩展到策略的概念: 基于决策理论对于待优化的目标函数最优值 (optimality) 的计算, 策略将所有的状态映射到一个最优的动作。

举例来说, 考虑一个典型的规划领域的问题: 以将一些特定的盒子放到一个指定的区域为目标来移动这些盒子。这个问题可以通过人工智能规划的手段来解决。进而考虑一个更现

实的场景：如果以将最多数目的盒子放在指定区域为目标，一些移动的动作（action）可能会失败，或者有不确定的副作用，而这些副作用来源于一些操作者不可控制的因素。在这种类型的环境中，因为依据操作者的不同，一系列的动作可以有許多不同的结果，这样的规划概念是不太合适的。相反，在本章中讨论的方法所使用的策略通过以下方式将状态映射到动作上：操作者的动作将对规划产生确定性的预期结果。动作的预期结果取决于有关概率性的结果和对目标的贡献（credit）的决策理论期望（decision-theoretic expectation）。马尔可夫决策过程的框架允许在线解决方案：通过模拟试验逐步地学习最优策略。此外，马尔可夫决策过程的框架允许基于计算资源的近似的解决方案，比如计算时间。最后，马尔可夫决策过程的框架允许对决策理论的策略质量与学习效果进行数值化度量。例如，学习策略可以通过对方程优化的贡献来度量，也可以通过达到某一个特定的性能所需要的计算量来度量。

本章将全面介绍现有文献中的基于马尔可夫决策过程模型的策略优化的方法。另外，本章还将介绍一些高级的内容，例如与强化学习相关的无先验知识学习。然后，任务和完成任务的算法将与环境（即马尔可夫决策过程）进行交互和试验，并根据反馈或者说奖励来获得如何对动作进行优化的知识。基于模型的设定（其中状态转换的规律和奖励的分布是完全已知的）通常是通过动态规划（Dynamic Programming, DP）技术进行描述的。然而，我们会看到，潜在的基础是非常相似的，并且出现了混合形式。

4

1.2 时序决策

强化学习是机器学习领域的一类算法，其目的在于：允许学习器（agent）学习如何在环境中动作，环境中仅有的反馈由标量奖励信号组成。强化学习不应该被当成一类特殊的学习方法，而是作为一个学习问题或范式。学习器的目标是长期最大化来自于环境的奖励信号。

学习器和环境之间的区别可能并不总是最直观的。我们将绘制一个基于控制的边界 [Sutton and Barto, 1998]。学习器不能控制的部分将被认为是环境的一部分。例如，虽然机器人学习器的电机可被认为是学习器的一部分，但这些电机在环境中的实际功能往往超出了学习器的控制。机器人学习器可以给电机向上或者向下的命令，但这些电机的实际动作往往被许多其他的因素影响。图 1.1 给出了一个学习器与环境相互作用的例子。图 1.1 显示了学习器和环境之间如何相互作用。学习器可以在每个状态中选择一个动作，学习器从环境中获得的感知是每一个动作后环境的状态加上在每一步获得的标量形式的奖励信号。离散模型需要从环境中获得每一个状态和动作的确切数字。描述的相互作用的方式是非常普遍的，只是作为离散符号谈论状态和动作。在本书的其余部分，我们将更加关注其相互作用，其中状态和动作拥有更多的结构，例如有两个蓝色盒子和一个白色盒子，而你站在一个蓝色盒子旁边。

环境	You are in state 65. You have 4 possible actions.
学习器	I'll take action 2.
环境	You have received a reward of 7 units. You are now in state 15.
	You have 2 possible actions.
学习器	I'll take action 1.
环境	You have received a reward of -4 units. You are now in state 65.
	You have 4 possible actions.
学习器	I'll take action 2.
环境	You have received a reward of 5 units. You are now in state 44.
	You have 5 possible actions.
...	...

图 1.1 从强化学习的角度，学习器与环境相互作用的例子

5

然而，这个图清楚地表明了时序决策的机制。

学习时序决策的几个重要方面将在本节介绍，之后我们将在下节形式化描述。

1.2.1 接近时序决策

有几种类型的算法可以用来处理时序决策的问题。在本书中，我们主要讨论与学习相关的主题，但是也讨论其他的一些题目。

第一个解决方案是基于编程（programming）的解决方案。原则上说，一个基于时序决策的智能系统可以编程处理所有可能的情况。对于每一个可能的状态，一个适当的或者最佳的动作需要来自先验知识的信息。然而，这个需求给智能系统的设计者和程序员带来了沉重的负担。在设计阶段，所有情况应该是可预见的，并在学习器中编制相关的程序。对于大多数有趣的问题来说，这是一个繁琐和几乎不可能的任务，只适用于可以完全建模的问题。在大多数现实问题中，由于问题的规模太大，或者系统中固有的不确定性，对问题进行完全建模是不可能的。一个简单的例子是机器人控制，其中的因素（例如灯光或温度）可以对机器人的相机和电机系统的动作有很大且不可预见的影响。此外，在问题发生变化的情况下，例如问题描述中出现了新的元素或者系统的动态发生变化，基于编程的解决方案将不再适用。因为基于编程的解决方案只能够工作在完全已知的且带有固定的概率分布的静态问题，这个解决方案是脆弱的。

第二种解决方案采用基于搜索和规划（search and planning）的时序决策。成功的国际象棋程序深蓝（Deep Blue）通过聪明的暴力搜索算法（brute force search algorithm）击败了世界冠军加里·卡斯帕罗夫（Gary Kasparov）[Schaeffer and Plaat, 1997]，而这个算法采用了国际象棋的动态模型并且调整到卡斯帕罗夫的棋风。当系统的动态是已知的，学习器可以通过搜索和规划，从当前的状态转移到下一个目标状态。然而，当动作的结果具有不确定性时，标准的搜索和规划算法就不适用了。可采纳的探索法（admissible heuristics）能够解决基于奖励的时序决策的问题，但是动作的概率性影响使问题的求解变得困难。概率规划算法是存在的，例如 [Kushmerick et al, 1995]，但概率规划算法的性能比不上确定性规划算法。一个额外的问题是，规划和搜索算法只关注于特定的起始状态和目标状态。相比之下，我们希望得到涵盖所有状态和奖励的策略。

第三个解决方案是学习，这将是本书的主题。学习在时序决策中有几个优点。首先，学习减轻了时序决策系统设计师的负担——在设计阶段，设计师不必考虑所有可能的情况。其次，学习可以应对系统的不确定性、基于奖励定义目标和不断变化的环境。最后，学习的目的是解决每一个状态的问题，而不仅仅是从一个状态转换到另一个状态的规划。此外，虽然环境的模型可以使用或学习，但是以计算最优策略为目标的模型并不是必需的，例如通过强化学习的方法。所有的知识都可以从与环境的互动中学习。

1.2.2 在线学习与离线学习

在本书中，我们讨论学习任务中的一个重要方面：在线学习和离线学习之间的区别。许多因素影响在线学习和离线学习之间的区别，例如控制的对象是否为现实世界的一个真实实体（如踢足球的机器人或工厂中的机械），或所有必要的信息是否可用。在线学习直接在问题的实例上进行学习。为了安全和快速地学习，离线学习使用环境的模拟器作为一种廉价的方式来获得训练样本。

在实际任务中直接学习控制器往往是不可能的。例如,在本章中的学习算法有时需要数以百万计的训练样本,收集这样数目的训练样本需要花费大量的时间。相反,使用环境的模拟器进行训练则快得多。此外,模拟器可以在任意情况下进行训练,包括在真实的系统中罕见的情况。再者,模拟器提供了一个相对“安全”的训练情况,即学习器可以探索和犯错误。在实际任务中获得负反馈来学习避免这些情况可能会导致破坏受控机器,这是不可接受的。通常情况下,学习器可以使用模拟来获得面向一个给定问题的合理策略。在实际任务中,学习器的动作是经过精密调试的。例如,模拟可能提供了学习一个合理的机器人控制器的手段,但在机器人电机和感知系统上引起变化的一些物理因素可能需要额外微调。模拟只是一个实际问题的模型,模拟与实际问题之间只存在细微的差异,学习能够弥补这种细微差异。然而,文献中提到的许多问题是游戏模拟和优化问题,这样的话就没有区别了。

1.2.3 贡献分配

时序决策的一个重要方面是:决定一个动作是“好”还是“坏”不能在短时间内完成。动作的适当性完全取决于学习器试图追求的目标。真正的问题是,有关目标的动作效果可以被大大地延迟。例如,国际象棋游戏的开局几步棋对整盘的输赢有很大的影响。然而,下完开局几步棋到获得赢棋奖励,中间可能还要下几十步棋。因为下开局几步棋的时候,还无法确定整盘棋的输赢。因此,决定开局几步棋对赢棋的贡献大小是一个困难的问题,而这个问题称为时间性贡献分配问题(temporal credit assignment problem)。在一个赢棋的国际象棋游戏中,虽然有几步棋可能不是最优的,甚至是坏的,但每一步都或多或少对最后的胜利做出了贡献。一个相关的问题是结构性贡献分配问题,其中的问题是根据代表学习器策略的结构分配反馈。例如,下棋策略可以由一个包含参数的结构表示(例如神经网络)。决定哪些参数需要更新形成了结构性贡献分配问题(structural credit assignment problem)。

7

1.2.4 探索-运用的平衡

如果我们知道问题的完整动态模型,现有的方法(例如动态优化)能够有效地计算出最优策略。然而,更一般的情况下(例如强化学习)我们无法获得这方面的知识,必须通过与环境交互和不断试错(trial-and-error)的方法学习正确的策略。学习器必须通过执行动作和感知动作结果的方式来探索环境,即对环境的影响和所获得的奖励。学习器得到的唯一反馈是奖励,但是学习器没有得到任何关于正确动作的信息。在某个时间点,学习器会拥有一个具备特定性能的策略。为了看看是否能改进策略,学习器有时必须尝试各种动作并检查这些动作的结果。因为有些动作也可能比目前的策略更差,这可能会导致更糟糕的性能。然而,如果没有尝试,学习器可能永远也找不到改进的方法。此外,如果环境是变化的,学习器必须探索以保持最新的策略。所以,为了学习,学习器必须探索。但是,为了更好的性能,学习器必须运用已经知道的知识。平衡这两个方面被称为探索-运用问题(exploration-exploitation problem)。

1.2.5 反馈、目标和性能

与监督学习相比,强化学习的学习系统从环境中获得的反馈是很少的。在监督学习中,训练集给出了每一个学习样本的正确输出。学习系统的性能可以通过正确答案的数量生成预测精度(predictive accuracy)进行度量。困难在于如何学到这种映射关系,以及学习到的映

射关系能否推广到新的、未知的例子。在无监督学习中,困难在于构建一个有价值的数据分区,而这个分区能自然而然地产生训练集的分类(class)。在强化学习中,只有一些关于性能的信息,以一个来自于环境的以标量数字形式存在的信号,反映出学习系统的性能。这种反馈系统是评估性的,而不是指导性的。因为环境只能提供有限的反馈信号,学习器必须更加努力地评估和改进动作。

关于反馈和性能的第二个方面是问题定义过程中的随机属性。在监督学习和无监督学习中,数据集通常被认为是静态的,即对于一个给定的数据集,学习器的性能通过这个数据集进行度量。学习者的学习样本源于一个固定的分布,即数据集。从强化学习的角度来看,数据集可以当成一个可移动的目标。学习过程是由当前的学习策略所驱动的,但随着时间的推移,学习策略也将随之改变。这意味着状态和奖励的分布也将随着时间不断改变。在机器学习中,学习样本的分布不断变化的问题称为概念漂移 [Maloof, 2003],而这个问题需要特殊的手段来解决。在强化学习中,学习样本的分布不断变化的问题是由探索来解决的:评估和策略改进之间的不断交互,再用自适应学习速率的方案。

反馈的第三个方面是以下问题:用于反馈的标量数字从哪里来?在许多时序决策的任务中,适当的奖励功能能够很自然地表达。对于有赢有输有平局的游戏,奖励函数是很容易定义的。在某些情况下,特别考虑了给予状态或动作奖励这个问题,而且它们的相对大小是很重要的。当学习器在最后获得一个小的正面奖励之前,遇到一个大的负面奖励,这使得正面奖励可能会蒙上阴影。当奖励函数与正确的目标一致,或者策略解决的是正确的问题,所有提出的学习问题都会有一些最优策略。在一些学习问题中,可以帮助带有奖励的学习器达到中间目标。这可能对需要非常长的动作序列的学习问题是有帮助的。

1.2.6 表达

学习时序决策的最重要的因素之一是表达(representation)。两个最重要问题是表达什么和如何表达。本章首先讨论了第一个问题:表达什么。可以或应该表达的关键组件是环境动态模型、奖励的分布、价值函数和策略。对于一些算法来说,例如经典的动态规划算法,所有的组成部分都是显式地存储在表中。演员-评论家方法(actor-critic method)对价值函数和策略进行了独立和明确的表示。然而,在大多数强化学习的算法中,价值函数只在策略来源于这个方程的时候才在线表达出来。在策略空间中搜索的方法隐式地表达价值函数。但是,当计算价值函数的价值时,策略显式地表达了出来。总的来说,省略某些元素的表达会影响一类算法的选择,以及这个算法的效率。

从下一章开始,本书将广泛地讨论各种结构如何表达这个问题。结构(例如策略)通过采用各种形式化的结构化知识表达方法,可以更紧凑地表达转换函数(transition function)和价值函数,这可以提供更有效的解决机制,扩展到更大的域。

1.3 正式的框架

本章 1.1 节描述的强化学习问题的元素可以通过马尔可夫决策过程的框架来正则化。在这一节中,我们将正式地描述学习的各个组成部分,包括状态、动作、策略,以及使用不同类型的最优准则确定的学习目标等。[Puterman, 1994; Boutilier et al, 1999] 深入地描述了马尔可夫决策过程。马尔可夫决策过程可以看成有限自动机(finite automata)的随机扩展,也可以看成增加了动作和奖励的马尔可夫过程(Markov process)。

虽然马尔可夫决策过程可能有极大的（甚至是不可数的）状态空间和动作空间，但本书仅仅讨论有限状态空间和有限动作空间的情况。在下一章中，我们将讨论连续空间。在后面的章节中，我们将遇到一阶逻辑设置的情况，在这种情况下，很自然地会讨论到无限空间。

1.3.1 马尔可夫决策过程

马尔可夫决策过程包括状态、动作、转换函数、奖励方程等。我们将逐个详细讨论这些概念。

1.3.1.1 状态

当状态空间的规模为 N ，环境状态集合 S 可以定义为有限集合 $\{s^1, \dots, s^N\}$ ，即 $|S| = N$ 。对于建模的问题来说，状态是所有信息中唯一的特征。例如，在国际象棋中，一个完整的棋盘由黑色方块和白色方块配置而成，这也是一种状态。在下一章中，我们将遇到使用特征来描述的状态。在这些情况下，有必要区分合法和非法状态，某些特征组合，可能不能描述在问题中实际存在的状态。在本章中，我们仅限于讨论离散的状态集合 S ，每一个状态由不同的符号表示，以及所有属于状态集合 S 的状态 s 都是合法的，即 $s \in S$ 。

1.3.1.2 动作

动作集合 A 通过有限集 $\{a^1, \dots, a^K\}$ 来定义。其中，动作空间的大小为 K ，即 $|A| = K$ 。动作可以用于控制系统的状态。能够应用于某一特定状态 $s \in S$ 的动作集合表示为 $A(s)$ ，其中 $A(s) \subseteq A$ 。在一些系统中，虽然不是动作集合 A 中的所有动作都能够运用于每一个状态，但是一般情况下，本书都默认动作集合 A 中的所有动作都能够运用于每一个状态，即，对于所有 $s \in S$ ， $A(s) = A$ 。事实上，在更多结构化的表达中，例如通过特征，有些动作无法运用于某些状态，而这些动作可以通过一个先决条件方程（precondition function）来建模： $S \times A \rightarrow \{\text{true}, \text{false}\}$ ，这个模型表达出动作 $a \in A$ 是否能够运用于状态 $s \in S$ 。

10

1.3.1.3 转换函数

通过将动作 $a \in A$ 运用于状态 $s \in S$ ，基于可能的转换集合的概率分布，学习系统能够完成从当前状态 s 到新状态 $s' \in S$ 的转换。转换函数 T 可以通过如下方式定义： $S \times A \times S \rightarrow [0, 1]$ ，即，作用于状态 s 的动作结束以后，其概率可以表示为 $T(s, a, s')$ 。所有的动作 a 和所有的状态 s 以及 s' 需要满足以下条件： $T(s, a, s') \geq 0$ 和 $T(s, a, s') \leq 1$ 。此外，所有的状态 s 和所有的动作 a 满足 $\sum_{s' \in S} T(s, a, s') = 1$ 。即， T 定义涵盖下面几个状态的概率分布。除先决条件方程之外，假设动作 a 不能运用于状态 s ，对于所有的状态 $s' \in S$ ，以下条件是不可能的： $T(s, a, s') = 0$ ^①。谈论动作发生的顺序，我们将定义一个离散的全局时钟 $t = 1, 2, \dots$ 。符号 s_t 表示在时间 t 的状态，而符号 s_{t+1} 表示在时间 $t+1$ 的状态。这使得在相互作用过程中按照时序发生的不同的状态（和动作）做比较成为可能。

如果一个动作的结果不依赖于以前的动作和历史的状况，而仅仅取决于当前状态，那么这个被控制的系统具有马尔可夫特性，即

$$P(s_{t+1} | s_t, a_t, s_{t-1}, a_{t-1}, \dots) = P(s_{t+1} | s_t, a_t) = T(s_t, a_t, s_{t+1})$$

马尔可夫动态（Markovian dynamics）的核心思想是当前的状态 s 提供足够的信息来做出最优决策，而当前状态 s 之前的状态和动作是不重要的。马尔可夫动态的核心思想的另一种表述方法是，下一个状态的概率分布与同一状态下的当前动作是一样的。更一般的模型

① 虽然这是相同的，但在一个状态中不适用的动作和与该动作转换的概率为零的显式区别是以这种方式丢失的。

可以通过 k -Markov 来描述。即, 最后 k 个状态足够使得马尔可夫实际上是 1-Markov。每个 k -Markov 问题都可以转化为一个等价的马尔可夫问题。马尔可夫特性是马尔可夫决策过程和更一般的模型 (例如 POMDP) 之间最典型的区别。

1.3.1.4 奖励方程

奖励方程^①指定在一个状态中的奖励, 或在一个状态中完成一个动作后的奖励。状态奖励方程可以定义为 $R:S \rightarrow \mathbb{R}$, 规定了状态中可以获得的奖励。然而, 还有两个其他定义, $R:S \times A \rightarrow \mathbb{R}$ 或者 $R:S \times A \times S \rightarrow \mathbb{R}$ 。第一个给在一个状态中执行的动作提供奖励, 第二个给状态之间的特定转换提供了奖励。因为我们通常需要备份起始状态和结束状态的数据, 虽然最后一个定义对无模型的算法 (model free algorithm) 是方便的 (见 1.7 节), 所有的定义都是可以互换的。在本书中, 我们将主要使用 $R(s,a,s')$ 。但是, 在更加方便的情况下会稍微偏离 $R(s,a,s')$ 。

奖励方程是马尔可夫决策过程最重要的部分, 因为奖励方程隐式地定义了学习的目标。例如, 在有情节的任务 (例如井字游戏 (Tic-Tac-Toe) 和国际象棋游戏) 中, 取得胜利的学习器中的所有状态可以被赋予正的奖励值, 取得失败结果的学习器中的所有状态可以被赋予负的奖励值, 最后取得和棋结果的学习器中的所有状态可以被赋予零的奖励值。该学习器的目标是达到积极的有价值的状态, 这意味着赢得比赛。因此, 奖励函数是用来给予受控制的系统 (即 MDP) 前进的方向的。通常, 奖励函数也将非零的奖励分配给非目标的状态, 这可以解释为学习而定义的子目标。

1.3.1.5 马尔可夫决策过程

将所有的元素合并一起导出了马尔可夫决策过程的定义, 这将是本书中所描述的大部分方法的基本模型。

定义 1.3.1 马尔可夫决策过程是一个由 4 个元素构成的元组 $\langle S,A,T,R \rangle$ 。其中, S 是一个包含所有状态的有限集合, A 是一个包含所有动作的有限集合, T 是一个定义为 $T:S \times A \times S \rightarrow [0,1]$ 的转换函数, R 是一个定义为 $R:S \times A \times S \rightarrow \mathbb{R}$ 的奖励方程。

转换函数 T 和奖励方程 R 一起定义了马尔可夫决策过程的模型。经常将马尔可夫决策过程描绘成一个状态转换图。其中, 图的节点对应状态, 图的 (有向) 边表示状态的转换。马尔可夫决策过程文献中常用的一个典型的例子是迷宫 [Matthews, 1922]。当即将达到迷宫出口时, 其奖励函数分配一个正的奖励。

有几个不同类型的系统可以通过马尔可夫决策过程的这个定义来建模。周期性任务拥有周期长度 (episode of length) 的概念, 在这个概念中, 学习的目标是将学习器从开始状态转换到一个目标状态。对于每个状态来说, 初始状态分布 $I:S \rightarrow [0,1]$ 给出了当前系统在此状态下开始的概率。根据所执行的动作, 从一个状态 s 开始, 系统通过一系列的状态前进。在周期性任务中, 有一个特定的子集 $G \subseteq S$, 这个子集表示过程结束时的目标状态区域, 而这个区域通常包含一些独特的奖励。此外, 任务可以进一步分为以下几种类型: 有限的固定范围的任务、未定义范围的任务和无限范围的任务。其中, 有限的固定范围的任务的每个阶段包含固定数目的步骤; 未定义范围的任务的每个阶段可以终止, 但是阶段的长度可以是任意的; 无限范围的任务包含无限的步骤, 学习系统永不停止。因此, 无限范围的任务通常称为

① 虽然我们在这里谈论的奖励通常带有积极的含义, 但回报函数只是给出一个标量反馈信号。这可以解释为否定 (惩罚) 或肯定 (奖励)。由于来自各种 MDP 文献, 使得回报函数的定义较为混乱。在运筹学文献中, 人们通常谈论成本函数, 而学习和优化的目标是最小化这个函数。

一个持续的任务。

12

周期性任务 (episodic task), 也就是所谓的目标状态 (goal state), 能够用定义 1.3.1 中的模型来建模。这通常是通过吸收状态 (absorbing state) 或终端状态 (terminal state) 来模拟的。例如, 每个动作的状态都是由概率 1 和奖励 0 转化为相同状态。正规的说, 对于吸收状态 s 来说, 对于所有的状态 $s \in S$ 和动作的 $a \in A$, 以下等式成立: $T(s, a, s') = 1$ 和 $R(s, a, s') = 0$ 。当进入一个吸收状态时, 进程将在一个新的启动状态下重新设置和重新启动。周期性任务和吸收状态可以用这种方式准确地模仿与连续任务相同的框架。

1.3.2 策略

给定一个马尔可夫决策过程 $\langle S, A, T, R \rangle$, 对于每一个状态 $s \in S$ 和每一个动作 $a \in A$ (或者 $a \in A(s)$) 来说, 策略是输出的可计算函数。形式上, 确定性的策略 π 是函数, 其定义为 $\pi: S \rightarrow A$ 。对于每一个状态 $s \in S$ 来说, 如果这个状态满足 $\pi(s, a) \geq 0$ 和 $\sum_{a \in A} \pi(s, a) = 1$, 那么定义如下的随机策略 (stochastic policy) 是可行的, $\pi: S \times A \rightarrow [0, 1]$ 。除非另有说明, 我们将在本书中假定确定性的策略。

马尔可夫决策过程的应用方法如下。首先, 从初始状态分布 I 产生一个起始状态的 s_0 。然后, 策略 π 建议的动作 $a_0 = \pi(s_0)$, 而这个动作将被执行。基于状态转移函数 T 和奖励函数 R , 转换到状态 s_1 , 那么概率为 $T(s_0, a_0, s_1)$ 且奖励 $r_0 = R(s_0, a_0, s_1)$ 将被接收。继续这个过程, 产生以下状态和动作: $s_0, a_0, r_0, s_1, a_1, r_1, s_2, a_2, \dots$ 。如果任务是周期性的, 这个过程将结束在状态 s_{goal} , 并从初始状态分布 I 中产生一个新的状态后重新启动。如果任务继续, 该序列的状态可以无限期执行。

策略是学习器的一部分, 而学习器的目的是控制环境, 而环境是用马尔可夫决策过程建模的。一个固定的策略是在马尔可夫决策过程中推导出一个静态转换, 而这个静态转换能够转换成马尔可夫系统[⊖] $\langle S', T' \rangle$, 满足以下条件: 当 $\pi(s) = a$ 时, $S' = S$ 和 $T'(s, s') = T(s, a, s')$ 。

1.3.3 最优准则和减量

在前面的章节中, 我们定义了环境 (即马尔可夫决策过程) 和学习器 (即控制元件或策略)。在我们讨论计算最优策略的算法之前, 我们必须定义这意味着什么。也就是说, 我们必须定义什么是最优模型。有两种方法看最优性。首先, 实际正在优化什么方面, 即学习器的目标是什么? 其次, 如何通过最优的方式优化目标。第一个方面与收集奖励有关, 并在本节中介绍。第二个方面与算法的效率和最优性有关, 这里简要地提及, 更广泛的内容将在 1.5 节以及后续章节中介绍。

13

在马尔可夫决策过程中学习的目标是收集奖励。如果学习器只关心即时奖励, 一个简单的最优准则将是优化 $E[r_t]$ 。然而, 有几种方法考虑到未来如何表现。马尔可夫决策过程最优模型基本上足以覆盖大部分的方法。它们与在 1.3.1 节中定义的任务类型密切相关。

$$\begin{array}{ccc} E\left[\sum_{t=0}^h r_t\right] & E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] & \lim_{h \rightarrow \infty} E\left[\frac{1}{h} \sum_{t=0}^h r_t\right] \\ \text{a)} & \text{b)} & \text{c)} \end{array}$$

图 1.2 a) 有限时域; b) 无限时域; c) 平均奖励

⊖ 换句话说, 如果 π 是固定的, 系统的行为与状态的平稳分布随机转换系统。

有限时域模型只需要一个有限长度 h 和规定学习器在这时域中应优化其预期回报，即下一个小时的步骤（见图 1.2a）。人们可以用两种方式来思考。一种方法是学习器可以在第一步采取 h 步最优行为，在这之后采取 $(h-1)$ 步最优行为，以此类推。另一种方法是学习器将始终采取 h 步最优行为，这称为滚动时域控制。然而，这个模型的问题是（最优）选择的时域长度 h 不总是已知的。

在无限时域模型中，将考虑长期奖励，但是根据接收时间的远近将在未来收到的奖励打折。这里使用阻尼系数 γ ， $0 \leq \gamma < 1$ （见图 1.2b）。请注意，在这种打折的情况下，越是后面获得的奖励，折扣就越大。此外，阻尼系数确保即使有无限的时域，得到的回报的总和是有限的。在阶段性任务（即任务的范围是有限的）中，阻尼系数是不需要的或可以等价地设置为 1。如果 $\gamma = 0$ ，学习器被称为“近视”，这意味着它只关心即时回报。阻尼系数可以用几种方法加以解释，如利率、再活一步的概率，或约束无限和数学技巧。打折的、无限的时域模型在数学上更方便，但在概念上类似于有限的时域模型。在本书中的大多数算法使用这种模型。

第三个最优模型是平均奖励模型，最大化的长期平均回报（见图 1.2c）。有时称为增益优化策略，在极限处，作为阻尼系数的方法 1 等于无限的时域贴现模型。长期（或无限）使用这一标准的一个棘手的问题是，我们不能区分两个策略，其中一个在初始阶段接收到了很多的奖励，另一个则没有。这种最初的回报差异隐藏在长期平均水平之中。这个问题可以使用偏置优化模型解决，其中的长期平均水平仍然是最优的，如果策略额外获得最初额外的奖励则其是首选。关于平均奖励强化学习的研究请见 [Mahadevan, 1996]。

最优准则之间的选择可以与学习问题相关联。如果周期长度是已知的，有限的时域模型是最好的。然而，通常这是未知的，或任务是连续的，则无限的时域模型更适合。[Koenig and Liu, 2002] 给出了马尔可夫决策过程中不同模型及其最优关系的概述。

在本节中，第二种最优模型与学习过程本身的最优性更一般的方面有关。在本书的其余部分，我们将遇到各种概念。我们将在这里简要总结三个重要的概念。

学习最终的结果可能是解释什么是学习最优性。首先要考虑的是学习器是否能够在原则上获得最优性能。对于一些算法有证明，说明这一点，但有些没有。换句话说，是否有一种方法可以确保学习过程达到一个全局最优，或仅仅是一个局部最优，甚至是性能在某个区间振荡？第二种最优模型与收敛到一个解决方案的速度有关。我们可以区分两种学习方法，通过寻找有多少相互作用是必要的，或每一个相互作用的计算量是多少。与之相关的，在一定的时间内其性能如何？在监督学习中最优准则通常在预测精度的方面定义，不同于最优的马尔可夫决策过程设置。此外，重要的是要看为了达到最佳的动作多少实验是必要的，甚至是允许的。例如，可能不会允许一个学习机器人或直升机在学习过程中犯许多错误。最后一种最优模型涉及相比一个最优策略，有多少奖励是没有包含在已学策略中。这通常称为一个学习系统的遗憾。

1.4 价值函数和贝尔曼方程

在前面的章节中我们定义了马尔可夫决策过程（MDP）和最优标准，有助于学习的最优策略。在这一节中，我们定义价值函数，这是一种连接最优准则和策略的方法。大多数针对 MDP 的学习算法通过学习价值函数来计算最优策略。一个价值函数表示一个估计，是在一个特定的状态（或是在该状态下采取的某一动作）对学习器的评估是否良好。良好的概念是通过最优准则（即在预期回报方面）来表达。价值函数为特定的策略所定义。

在策略 π 下的状态 S 的值, 表示为 $V^\pi(s)$, 是预期收益。本节中, 我们将使用无限的时域贴现模型, 可以表示为^①: [15]

$$V^\pi(s) = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s \right\} \quad (1.1)$$

一个类似的状态-动作价值函数 $Q: S \times A \rightarrow \mathbb{R}$ 可以定义为预期收益从状态 s 出发, 采取动作 a , 此后跟随以下策略 π :

$$Q^\pi(s, a) = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s, a_t = a \right\}$$

价值函数的一个基本属性是这些函数满足一定的递归性。在公式 (1.1) 中表达的任何策略 π 和任何状态 s 可以根据所谓的贝尔曼方程递归地定义 [Bellman, 1957]:

$$\begin{aligned} V^\pi(s) &= E_\pi \{ r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots \mid s_t = s \} \\ &= E_\pi \{ r_t + \gamma V^\pi(s_{t+1}) \mid s_t = s \} \\ &= \sum_{s'} T(s, \pi(s), s') (R(s, a, s') + \gamma V^\pi(s')) \end{aligned} \quad (1.2)$$

以上的公式表示状态的期望值是指在可能的转换概率加权的下一个状态的即时奖励和价值, 外加一个阻尼系数。 V^π 是这组方程的唯一解。注意多个策略可以有相同的价值功能, 但对于一个给定的策略 π , V^π 是独一无二的。

对于任何给定的马尔可夫决策过程的目标是找到一个最优策略, 即获得最多的奖励策略。这意味着对所有状态 $s \in S$ 最大化公式 (1.1) 的价值函数。最优策略表示为 π^* , 并对所有策略的 π 满足以下条件 $V^{\pi^*}(s) \geq V^\pi(s)$ 。可以证明最优解决方案 $V^* = V^{\pi^*}$ 满足如下方程:

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V^*(s')) \quad (1.3)$$

这个表达式称为贝尔曼最优方程 (Bellman optimality equation)。一个最优策略下的状态值必须等于在该状态下的最优行为的预期回报。为了选择了最优状态的价值函数 V^* 的最优行为, 可以应用以下规则:

$$\pi^*(s) = \arg \max_a \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V^*(s')) \quad (1.4)$$

我们称这种策略为贪婪策略, 表示为 $\pi_{\text{greedy}}(V)$, 因为贪婪地选择运用价值函数 V 的最优行为。类似的最优状态-动作值 (optimal state-action value) 可以计算为:

$$Q^*(s, a) = \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma \max_{a'} Q^*(s', a'))$$

Q 函数是有用的因为它们对不同的选择进行加权求和 (weighted summation), 例如公式 (1.4), 未必使用转换函数。不需要向前的推理步骤 (forward-reasoning step) 来在状态中计算最优行为 (optimal action)。这是无模型方法 (model-free approach) 的原因, 即如果 T 和 R 是未知的, 学习的不是 V 函数, 而是 Q 函数。 Q^* 和 V^* 之间的关系是由以下公式确定的:

$$Q^*(s, a) = \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V^*(s')) \quad (1.5)$$

① 注意, 我们使用 E_π 期望值下的策略 π 。

$$V^*(s) = \max_a Q^*(s, a) \quad (1.6)$$

现在, 类似于公式 (1.4), 选择最优行为可以简单地计算为:

$$\pi^*(s) = \arg \max_a Q^*(s, a) \quad (1.7)$$

也就是说, 最优行为是基于可能产生的下一个状态所采取的动作, 其有最高的预期效用。类似于公式 (1.4), 可以在 Q 上定义一个贪婪策略 $\pi_{\text{greedy}}(Q)$ 。与 $\pi_{\text{greedy}}(V)$ 不同的是, 没有必要参考马尔可夫决策过程模型, 有 Q 函数就够了。

1.5 求解马尔可夫决策过程

现在我们已经定义了马尔可夫决策过程策略、价值函数和最优准则, 是时候考虑如何计算最优策略了。求解给定的马尔可夫决策过程意味着计算最优策略 (optimal policy) π^* 。已经开发了几个方面的用于此目的算法。最重要的区别是基于模型的算法和无模型的算法。

基于模型的算法一般称为动态规划。这些算法的基本假设是一个马尔可夫决策过程的模型是事先已知的, 并且可以用来计算价值函数和应用贝尔曼方程的策略 (见公式 (1.3))。大多数方法的目的是计算状态价值函数, 在模型中, 状态价值函数用于最优行为的选择。在本章中, 我们将重点放在计算价值函数和策略的迭代过程。

无模型算法 (一般称为强化学习) 不依赖于一个完美模型的可用。相反, 它们依赖于与环境的相互作用, 即一个策略的模拟, 从而产生样本的状态转换和奖励。然后, 这些样本可以用来估计状态-动作价值函数 (state-action value function)。因为 MDP 模型是未知的, 只能通过探索 MDP 获取信息。这个方法自然地推导出探索-运用之间的平衡, 以获得一个最优策略。在基于模型的强化学习, 学习器不具有一个先验的环境模型, 而是边学习边估测模型。在推导了一个合理的环境模型后, 学习器可以应用动态规划算法来计算一个策略。

一个非常重要的机制叫作广义策略迭代 (Generalized Policy Iteration, GPI) 原则, 如图 1.3 所示的方法描述了该原则的方法。这个原则由两个相互作用的过程组成。策略评估步骤 (policy evaluation step) 估计目前的策略 π , 也就是说, 有几种方法可以用于计算 V^π 。在基于模型的算法中, 可以使用该模型来直接计算, 或采用迭代近似 (iteratively approximate)。在无模型算法中, 可以模拟策略并从采样的执行痕迹估计其效用。这一步的主要目的是收集关于计算第二步 (策略改进步骤) 的策略的信息。在第二步中, 对每个状态中的动作值进行评估以找到可能的改进, 即, 特定的状态下其他可能的好于目前策略的动作。利用 V^π 中的信息, 这一步从目前策略 π 计算改进策略 π' 。评估步骤和改进步骤可以以不同的方式实现, 并在几个不同的方式中交错。底线是: 有一个策略推动价值学习 (value learning), 即决定价值函数, 而反过来策略可使用一个价值函数以选择良好的动作。请注意, 也可以有一个隐式表示的策略, 这意味着只存储价值函数。在需要的时候, 每个状态的策略是基于价值函数即时计算的。在无模型的算法中, 这是常见的做法, 详见 1.7 节。反之亦然, 也可以在一个明确的策略表示的背景下, 有隐含的价值函数的表示。另一个有趣的方面是, 一般来说, 价值函数不必是完全准确的。在许多情况下, 这就足以区别是次优动作 (suboptimal action) 和最优的动作之间 (optimal actions), 这样小的误差值没有影响策略的最优性。这在近似和抽象方法中很重要。

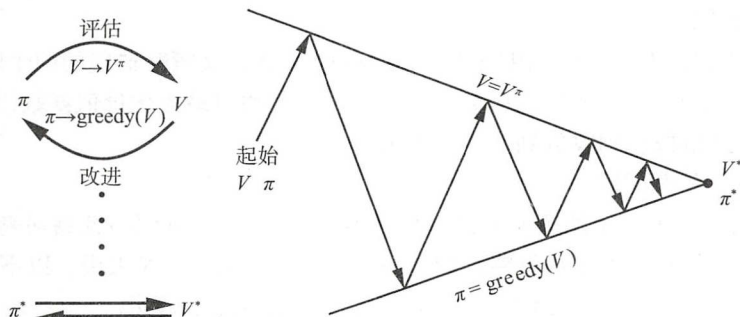


图 1.3 在 1.5 节中的算法可以看作广义策略迭代 [Sutton and Barto, 1998]。策略评估步骤估计策略的性能 V^π 。基于 V^π 的估计值，策略的改进步骤改进策略 π 。b) 价值函数和策略的逐渐收敛至最优版本

一个强化学习问题的规划

马尔可夫决策过程的形式化是决策理论规划 (decision-theoretic planning) 的一般形式，这需要可以形式化的标准的 (确定性的) 规划问题。在原则上，在本章中的所有算法都可以用于这些规划问题。为了解决马尔可夫决策过程框架中的规划问题，必须指定目标和奖励。我们可以假设，伴随着一个先决条件的转换函数 T 是给定的。在规划中，我们有一个目标函数 $G: S \rightarrow \{\text{true}, \text{false}\}$ ，其定义了目标状态。规划任务 (planning task) 是计算一个动作序列 $a_t, a_{t+1}, \dots, a_{t+n}$ ，应用这个序列可以从起始状态到达目标状态 $s \in G$ 。假定所有的转换是确定性的，即，对于所有状态 $s \in S$ 和所有动作 $a \in A$ 而言，只有一个满足 $T(s, a, s') = 1$ 条件的状态 $s' \in S$ 存在。 G 中的所有状态都假定为可吸收 (absorbing)。唯一剩下的是指定的奖励函数。我们可以以如下方式指定奖励函数：当达到目标状态时，将给予一个正面的奖励，否则不给予奖励。

$$R(s_t, a_t, s_{t+1}) = \begin{cases} 1, & s_t \notin G \text{ 且 } s_{t+1} \in G \\ 0, & \text{其他} \end{cases}$$

现在，取决于是否转换函数和奖励函数为学习器所知，无论是基于模型的学习或无模型学习都可以求解该规划任务。与典型规划的区别是学会的策略将适用于所有状态。

1.6 动态规划：基于模型的解决方案

动态规划是指可以计算在一个完善的环境模型下的最优策略的一类算法。假设一个模型难以用于许多应用。然而，从理论的角度以及算法的角度来看，我们会看到动态规划算法适用于此情景，因为其定义了基本运算机制 (当无模型可用时) 也能使用。本节中的方法都是假设一个标准的 MDP $\langle S, A, T, R \rangle$ ，其中状态和动作集是有限的、离散的从而可以存储在表中。此外，假定转换、奖励和价值函数分开存储值所有状态和动作。

1.6.1 基本的动态规划算法

两个核心的动态规划方法是策略迭代 (policy iteration) [Howard, 1960] 和数值迭代 (value iteration) [贝尔曼, 1957]。首先，广义策略迭代机制明显分为两个步骤，其中第二个步骤代表一个策略评估和改进的紧密集成 (tight integration)。我们将依次讨论这两种算法。

1.6.1.1 策略迭代

策略迭代 (Policy Iteration, PI) [Howard, 1960]: 在广义策略迭代的两个阶段之间循环。策略评估阶段计算当前策略的价值函数, 策略改进阶段通过最大化价值函数以计算一个改进的策略。重复上述阶段直到收敛到一个最优策略。

1. 策略评估: 预测问题

第一步是找到一个固定的策略 π 的价值函数 V^π 。这就是所谓的预测问题。这是一个完整问题的一部分, 计算一个最优策略。如前所述, 对于所有 $s \in S$ 来说, 以下等式成立:

$$V^\pi(s) = \sum_{s' \in S} T(s, \pi(s), s') (R(s, \pi(s), s') + \gamma V^\pi(s')) \quad (1.8)$$

如果系统的动力学是已知的, 即马尔可夫决策过程的模型是给定的。然后, 对于每一个 $s \in S$ 来说 V^π 的值 (即系统 $|S|$ 的等式 $|S|$ 的值) 是未知的。这可以通过线性规划 (Linear Programming, LP) 来解决。然而, 可能存在一个迭代过程, 并且在动态规划和强化学习中非常常见。贝尔曼方程转化为一个更新规则, 通过“在未来多看一步”的策略, 这个规则将当前价值函数从 V_k^π 更新到 V_{k+1}^π , 从而一步延长规划的时域 (planning horizon):

$$\begin{aligned} V_{k+1}^\pi(s) &= E_\pi \{r_t + \gamma V_k^\pi(s_{t+1}) | s_t = s\} \\ &= \sum_{s'} T(s, \pi(s), s') (R(s, \pi(s), s') + \gamma V_k^\pi(s')) \end{aligned} \quad (1.9)$$

当 k 趋于无穷大时, V_k^π 的近似值序列可以收敛。为了收敛, 对于每一个循环中的每一个状态 $s \in S$, 应用更新规则。由一个新的价值取代了旧的价值, 而这个新的值是基于可能的继承状态 (possible successor state) 的预期值、中间奖励和加权的转换概率。因为是基于该状态所有可能的转换的, 这个操作称为完整备份 (full backup)。通过在状态空间 (例如价值函数) 中的任意实值函数 φ 上定义一个备份操作 B^π , 可以给出更一般的公式。

20

$$(B^\pi \varphi)(s) = \sum_{s' \in S} T(s, \pi(s), s') (R(s, \pi(s), s') + \gamma \varphi(s')) \quad (1.10)$$

当 $V^\pi = B^\pi V^\pi$ 时, 固定策略 π 的价值函数 V^π 满足这个备份操作的固定点。在一个有用的特殊情况下, 相对于一个固定的动作 a 定义该备份操作:

$$(B^a \varphi)(s) = R(s) + \gamma \sum_{s' \in S} T(s, a, s') \varphi(s')$$

现在求解预测问题的线性规划可以总结如下。对于所有状态而言, 计算 V^π 可以通过求解贝尔曼方程完成 (见公式 (1.3))。对于所有的 a 和 s 来说, 在 $V(s) \geq (B^a V)(s)$ 的条件下, 最优价值函数 V^* 可以通过求解线性规划问题计算 $V^* = \arg \max_V \sum_s V(s)$ 来求解。

2. 策略改进

现在, 我们知道了策略 π 的价值函数 V^π , 可以作为策略评估步骤的结果, 我们可以尝试改进策略。首先, 我们确定使用的所有动作值:

$$Q^\pi(s, a) = E_\pi \{r_t + \gamma V^\pi(s_{t+1}) | s_t = s, a_t = a\} \quad (1.11)$$

$$= \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma V^\pi(s')) \quad (1.12)$$

对于一些 $a \in A$ 来说, 如果 $Q^\pi(s, a)$ 大于 $V^\pi(s)$, 相比处理当前 $\pi(s)$, 我们最好寻求一个新的动作 a 。换言之, 在一个特定的状态下, 我们可以通过选择一个不同的、更好的动作, 来改善当前策略。事实上, 我们可以评估所有状态的所有动作, 并在所有状态中选择最

好的动作。也就是说，通过在每一个状态，基于当前的价值函数 V^π ，我们可以计算贪婪策略 π' 来选择最优行为：

$$\begin{aligned}\pi'(s) &= \arg \max_a Q^\pi(s, a) \\ &= \arg \max_a E\{r_t + \gamma V^\pi(s_{t+1}) | s_t = s, a_t = a\} \\ &= \arg \max_a \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma V^\pi(s'))\end{aligned}\quad (1.13)$$

对原有策略的价值函数，通过贪婪地选择最优行为来计算改进策略，称为所谓的策略改进。如果策略不能以这种方式得到改进，这意味着策略已经是最优的，即策略的价值函数满足的最优价值函数——贝尔曼方程。以类似的方式，通过混合动作概率和期望算子，也可以执行这些步骤的随机策略。

总的来说，策略迭代 [Howard, 1960] 可以从任意的初始策略 π_0 开始。然后，一个迭代序列如下所示，其中当前策略进行评估后再改进。第一步是策略评估，迭代地使用公式 (1.9) 来计算 V^{π_k} 。第二步是策略改进，采用 V^{π_k} 计算 π_{k+1} 。对于每一个状态来说，使用公式 (1.4)，最优行为是确定的。如果对于所有状态 s ， $\pi_{k+1}(s) = \pi_k(s)$ ，则策略稳定，策略迭代算法可以停止。策略迭代生成一个交替的策略和价值函数

$$\pi_0 \rightarrow V^{\pi_0} \rightarrow \pi_1 \rightarrow V^{\pi_1} \rightarrow \pi_2 \rightarrow V^{\pi_2} \rightarrow \pi_3 \rightarrow V^{\pi_3} \rightarrow \dots \rightarrow \pi^*$$

完整的算法可以在算法 1 中找到。

```

Require:  $V(s) \in \mathbb{R}$  and  $\pi(s) \in A(s)$  arbitrarily for all  $s \in S$ 
{POLICY EVALUATION}
repeat
   $\Delta := 0$ 
  for each  $s \in S$  do
     $v := V^\pi(s)$ 
     $V(s) := \sum_{s'} T(s, \pi(s), s') (R(s, \pi(s), s') + \gamma V(s'))$ 
     $\Delta := \max(\Delta, |v - V(s)|)$ 
until  $\Delta < \sigma$ 
{POLICY IMPROVEMENT}
policy-stable := true
for each  $s \in S$  do
   $b := \pi(s)$ 
   $\pi(s) := \arg \max_a \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma V(s'))$ 
  if  $b \neq \pi(s)$  then policy-stable := false
if policy-stable then stop; else go to POLICY EVALUATION

```

算法 1 策略迭代 [Howard, 1960]

对于有限马尔可夫决策过程来说，即状态和动作空间是有限的，策略迭代收敛于有限次的迭代之后。除非 $\pi_k = \pi^*$ ，每一个策略 π_{k+1} 是一个比 π_k 严格更好的策略。在这种情况下，算法将停止运行。因为对于一个有限的马尔可夫决策过程，不同策略的数量是有限的，在有限的时间内收敛策略迭代。在实践中，通常在少量迭代后收敛。虽然对于一个给定的马尔可夫决策过程，在有限的时间内，策略迭代计算最优策略，但是计算是相对低效的。特别是第一步（策略评估步骤），计算上是昂贵的。对于价值函数的所有中间策略 $\pi_0, \dots, \pi_k, \dots, \pi^*$ 的计算涉及通过每次迭代的完整状态空间的多个扫描（multiple sweeps）。循环的上下界是未知的

[Littman et al, 1995], 并依赖于马尔可夫决策过程的转换结构, 但是实际上在少量的循环以后收敛。

```

Require: initialize  $V$  arbitrarily (e.g.  $V(s) := 0, \forall s \in S$ )
repeat
   $\Delta := 0$ 
  for each  $s \in S$  do
     $v := V(s)$ 
    for each  $a \in A(s)$  do
       $Q(s, a) := \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma V(s'))$ 
     $V(s) := \max_a Q(s, a)$ 
     $\Delta := \max(\Delta, |v - V(s)|)$ 
until  $\Delta < \sigma$ 

```

算法 2 值迭代 [Bellman, 1957]

1.6.1.2 价值迭代

策略迭代算法完全分离了评估阶段和改进阶段。在评估步骤中, 价值函数必须在极限中计算。然而, 没有必要等待完全收敛, 但是有可能提前停止评估, 并且在评估的基础上改进策略。终止评估步骤的极值点 (extreme point) 是值迭代算法 [Bellman, 1957]。一次迭代后, 评估就中断了。事实上, 它立即将策略改进的步骤融合到迭代之中, 从而纯粹专注于直接估计价值函数。即时计算必要更新。从本质上说, 这个方法结合了一个删减版本的策略评估步骤与策略改进步骤。这个方法的本质是将公式 (1.3) 变成一个更新规则:

$$V_{t+1}(s) = \max_a \sum_{s'} T(s, a, s') (R(s, a, s') + \gamma V_t(s')) \quad (1.14)$$

$$= \max_a Q_{t+1}(s, a) \quad (1.15)$$

如算法 2 所示, 利用公式 (1.14) 和公式 (1.15), 该值迭代算法可以表述为: 从所有状态的价值函数 V_0 开始, 根据公式 (1.14), 迭代更新每个状态的值来获取下一个价值函数 V_t ($t = 1, 2, 3, \dots$)。这个方法产生以下的价值函数序列:

$$V_0 \rightarrow V_1 \rightarrow V_2 \rightarrow V_3 \rightarrow V_4 \rightarrow V_5 \rightarrow V_6 \rightarrow V_7 \rightarrow \dots \rightarrow V^*$$

事实上, 这样的计算方法也会产生中间的 Q 价值函数的序列:

$$V_0 \rightarrow Q_1 \rightarrow V_1 \rightarrow Q_2 \rightarrow V_2 \rightarrow Q_3 \rightarrow V_3 \rightarrow Q_4 \rightarrow V_4 \rightarrow \dots \rightarrow V^*$$

值迭代保证向极限 V^* 收敛, 即贝尔曼最优方程 (公式 (1.3)) 对于每个状态来说都是成立的。对于所有状态 $s \in S$ 来说, 一个确定策略的 π 可以通过公式 (1.4) 来计算。如果我们使用上一节中使用的相同的通用备份算子机制 (general backup operator mechanism), 我们可以用以下方式来定义值迭代:

$$(B^* \varphi)(s) = \max_a \sum_{s' \in S} T(s, a, s') \{R(s, a, s') + \gamma \varphi(s')\} \quad (1.16)$$

备份算子 B^* 可以作为价值函数的收缩映射 (contraction mapping)。如果我们让 π^* 表示最优策略, V^* 表示价值函数, 那么以下等式是成立的: 当 $(B^* V)(s) = \max_a (B^a V)(s)$ 时, $V^* = B^* V^*$ 。如果定义了 $Q^*(s, a) = B^a V^*$, 那么 $\pi^*(s) = \pi_{\text{greedy}}(V^*)(s) = \operatorname{argmax}_a Q^*(s, a)$ 。也就是说, 该算法从任意一个价值函数值 V_0 之后, 就开始以下循环: $V_{t+1} = B^* V_t$, 直到 $\|V_t - V^*\| < \epsilon$ 。即, 直到随后的价值函数逼近的距离足够小。

1.6.2 高效的动态规划算法

策略迭代算法和值迭代算法能够看成生产动态规划方法的频谱。此频谱范围从完整的分离评估和改进步骤，到完整的集成这些步骤。显然，在极值点之间，算法变化的空间是很大的。让我们先考虑极值点的计算复杂度。

计算复杂度

值迭代法通过产生最优价值函数的连续逼近 (successive approximation) 进行工作。每一次迭代可以在 $O(|A||S|^2)$ 步骤完成。如果 T 是稀疏的，能够更快。然而，在阻尼系数中，迭代次数可能会呈现指数级的增长 [Bertsekas and Tsitsiklis, 1996]。事实是，一个更大的 γ 意味着必须考虑到一个更长的未来奖励序列。因此，需要一个更多的值迭代步骤，因为每步只考虑扩展 V 中的一步。值迭代的复杂性在动作的数量上是线性的 (linear)，在状态的数目上是二次的 (quadratic)。但是，通常情况下的转换矩阵 (transition matrix) 是稀疏的。在实践中，策略迭代收敛速度快得多，但每一个评估步骤是昂贵的。每次迭代的计算复杂度为 $O(|A||S|_2 + |S|_3)$ ，这个计算复杂度可以迅速增长。最坏的情况下，在迭代的次数的上下界是未知的 [Littman et al, 1995]。线性规划是一种常用的工具，也可以用于评估。在一般情况下，当问题的规模增长时，迭代的次数和值备份的数量可以迅速增长得非常大。仅仅执行一次完全的扫描 (full sweep)，游戏 (例如五子棋和象棋) 就拥有很大规模的状态空间。在这一节中，我们将描述动态规划方法的一些高效的改进版本。求解问题的复杂度详见文献 [Littman et al, 1995; Bertsekas and Tsitsiklis, 1996; Boutilier et al, 1999]。

24

动态规划 (DP) 的效率可以大致通过两种方式提高。第一种方式是与广义策略迭代过程的评估步骤和改进步骤更紧密地结合。我们将在下一节简要讨论这个问题。第二种方式是采用 (启发式) 搜索算法，并与动态规划算法结合起来。例如，使用搜索作为一个探索机制 (exploration mechanism) 可以突出的状态空间的重要组成部分。而且，值备份 (value backup) 可以集中在这些部分。这些方法的潜在机制将在 1.6.2.2 节讨论。

1.6.2.1 更新的风格

动态规划算法的完全备份更新能够通过几种方式完成。在每一个步骤中，我们已经假定在描述的算法中，在内存中保存一个旧的价值函数和一个新的价值函数。根据旧的信息，每一个更新都存储在新的列表中。这就是所谓的同步或 Jacobi- 风格更新 [Sutton and Barto, 1998]。这对于算法的解释和收敛的理论证明都是有用的。然而，有两个更常见的方式更新。一种方式是可以保存单个表，并且直接在那里更新，这就是所谓的本地更新 (in-place updating) [Sutton and Barto, 1998] 或 Gauss-Seidel 法 [Bertsekas and Tsitsiklis, 1996]。通常加速收敛，因为通过扫描的方式更新期间，一些更新使用新近更新的其他状态的值。另一种方式称为异步更新 (asynchronous updating)，这是一个扩展的本地更新，但这里的更新可以以任何顺序进行。它的一个优点是更新可能会在状态 - 动作空间中呈现分布不均的状态，将更多的更新给予这个空间中更重要的部分。对于所有这些方法的收敛性，可以证明在一般条件下，值被更新的次数是无限的，但具有的频率是有限的。

修改后的策略迭代

修改后的策略迭代 (Modified Policy Iteration, MPI) 在值迭代和策略迭代之间摇摆 [Puterman and Shin, 1978]。修改后的策略迭代保持广义策略迭代的两个单独的步骤，但两个步骤不一定在极限内计算。这里的关键是，对于策略的改进来说，为了改进策略，并不需

要一个精确的评估策略。例如，在策略评估步骤之后，策略改进步骤可以是近似的。在一般情况下，这两个步骤可以通过不同的方式相互独立地进行。例如，除了反复应用来自于公式 (1.15) 的贝尔曼更新规则之外，可以采用样本采集步骤执行策略评估步骤，例如蒙特卡罗估计方法 (Monte Carlo estimation) [Sutton and Barto, 1998]。这些具有估计和改进混合的一般形式能够被图 1.3 中描述的一般化的策略迭代机制所抓取。策略迭代和值迭代都是修改后的策略迭代的极端情况，而修改后的策略迭代是异步更新的一般方法。

1.6.2.2 启发式搜索

在许多实际问题中，在状态空间 (state space) 中，只有一小部分状态与问题相关，并能从某个状态 s 到达目标状态。这启发了很多算法可侧重于计算从开始状态 s 到发现最优策略的相关状态。这些算法通常显示良好的随时动作。即，这些算法快速产生良好的或者合理的策略，随后逐渐改进这些算法。此外，这些算法可以看作各种方式实现的异步动态规划 (asynchronous DP)。

1. 信封状态和边缘状态

异步方法的一种形式是 PLEXUS 系统 [Dean et al, 1995]。这些方法都是基于目标的奖励函数，即周期性的任务，其中只有目标状态得到正面的奖励。这些方法从一个马尔可夫决策过程的近似版本开始，这个版本并不包括完整的状态空间。这个马尔可夫决策过程的缩写版本称为信封 (envelope)，包括学习器的当前状态和目标状态。一个特殊的 OUT 状态表示信封以外的所有状态。最初的信封是由一个前向搜索 (forward search) 所构成，直到找到一个目标状态。这个信封可以通过考虑信封外高概率可实现的状态来进行扩展。直观的想法是把所有可能达到目标的状态都包括在信封里。一旦构建了信封，策略就会通过策略迭代计算出来。如果在任何时候，学习器离开了信封，这个学习器就必须通过扩展信封来重新规划。学习和规划的结合仍然使用策略迭代，但是在一个小得多 (且与目标更相关) 的状态空间中执行。[Tash and Russell, 1994] 中提出的相关的方法也考虑了基于目标的任务。然而，除了单一的 DUT 状态之外，在信封边缘保留了一些状态的边缘。然后，用启发式的方法来估计其他状态的价值。当计算一个信封的策略时，所有的边缘状态 (fringe state) 成为吸收状态，通过启发式设置这些状态值。随着时间的推移，边缘状态的启发式值收敛到这些状态的最优值。与前面的方法类似，LAO* 算法也在扩展阶段和策略生成阶段之间交替进行 [Zilberstein, 2001]。在信封外保留了一些边缘状态，这样扩展就可以大于 [Dean et al, 1995] 中的信封方法。LAO* 算法的目的是将经典的搜索算法 AO* 扩展到循环领域 (cyclic domain)，例如马尔可夫决策过程，AO* 算法详见 [Russell and Norvig, 2003]。

2. 动态规划之中的搜索与计划

实时动态规划 (Real-Time Dynamic DP, RTDP) 将动态规划与前向搜索相结合 [Barto et al, 1995]。这个方法可替代值迭代。在每次迭代中，只有状态空间中值的子集得到备份。通过使用一种可获得的启发式函数 (admissible heuristic function) 作为初始价值函数 (initial value function) 来模拟贪婪策略，实时动态规划从一个随机选择的状态到目标状态不断地进行试验。然后，实时动态规划只在这些测试中完全备份值，这样备份就会集中在状态空间的相关部分上。这个方法后来被扩展到有标签的实时动态规划中 [Bonet and Geffner, 2003b]，一些状态标记为已解，这意味着这些算法的值已经收敛了。此外，最近扩展到有界的实时动态规划，其保持了最优价值函数的下界和上界 [McMahan et al, 2005]。最近的其他方法是集



中的动态规划 [Ferguson and Stentz, 2004] 和启发式的搜索 - 动态规划 [Bonet and Geffner, 2003a]。

1.7 强化学习：无模型的解决方案

假设有一个（完美的）模型可用，前一节已经回顾了几个计算马尔可夫决策过程的最优策略的方法。当这样的模式不可用时，强化学习主要关注如何获得最优策略。与马尔可夫决策过程相关的强化学习关注于逼近（approximation）和不完整信息（incomplete information），以及对采样和探索的需求。与前一节中讨论的算法对比，无模型的方法不依赖于先验的转换和奖励可用性模型，即马尔可夫决策过程的模型。模型的缺乏使得需要采样马尔可夫决策过程以获取未知模型的统计知识。许多无模型强化学习技术通过动作探索，从而估计状态值和状态动作价值函数相同的基于模型的技术。本节将回顾无模型的方法以及几种有效的扩展。

首先，在无模型的情况下，仍然有两个选项以供选择。第一个选项是首先从与环境的交互中学习转换和奖励模式。之后，如果该模型（近似或足够）正确，在上一节中的所有的动态规划方法都可以应用。这种类型的学习称为间接强化学习（indirect RL）或基于模型的强化学习（model-based RL）。第二个选项称为直接强化学习，是直接估计动作值，甚至没有估计马尔可夫决策过程模型。此外，这两种形式也可以混合使用。例如，仍然可以为动作值做无模型的估计（model-free estimation），但使用一个近似模型加快价值学习，另外，完全备份值（见 1.7.3 节）。然而，大多数无模型的方法专注于直接估计（动作）值。

其次，要做的是如何利用时间贡献分配问题。如果很晚才察觉一个特定的动作的真正影响，那么很难评估该动作的效用。一种可能性是等到“结束”（例如一个周期的末尾）惩罚或奖励所采取的具体动作。然而，这将需要大量内存并且经常有正在进行的任务，事先不知其是否或何时会“结束”。相反，人们可以使用类似的机制，基于即时奖励和下一个状态的估计（贴现）值在值迭代中调整状态的估计值。这通常称为时序差分学习，这是本节中无模型方法的通用机制。与动态规划方法的更新规则的主要区别（如公式（1.14））是现在在更新规则中不能出现转移函数 T 和奖励函数 R 。这样的与环境交互并在每次经历后更新它们的估计的算法通用类称为在线强化学习。

[27]

一般在线强化学习的通用模板如算法 3 所示。它显示了一个交互回路，其中学习器基于当前状态选择一个动作（以任何方式），以结果状态和相关奖励的形式得到反馈，之后它更新其存储在 \tilde{V} 和 \tilde{Q} 中的估计值，可能统计 \tilde{T} 和 \tilde{R} （在一些间接学习的案例中）。该动作的选择基于当前状态 s 和价值函数（不是 Q 就是 V ）。为了解决探索 - 运用的问题，通常使用一个单独的探索机制，其确保有时采取（运用）最好的动作（根据目前的动作值估计），但有时选择（探索）一个不同的动作。探索不同选择（从随机选择到复杂选择）的示例详见 1.7.3 节和下一部分。

探索

无模型算法的一个重要方面是需要探索。由于该模型是未知的，学习器必须尝试不同的动作来看到它们的结果。一个学习算法需要平衡探索和运用，即为了获得大量奖励，学习器需要利用有关良好的动作的目的的知识，尽管有时必须尝试不同的动作来探索寻找可能更好的动作环境。最基本的搜索策略是 ϵ -贪婪策略，即学习器目前处于最佳动作的概率是 $(1-\epsilon)$ ，而（随机选择）其他动作的概率是 ϵ 。有很多种探索方式，见 [Wiering, 1999；



Reynolds, 2002 ; Ratitch, 2005], 1.7.3 节中我们将看到一些例子。另外一个方法（通常用于与本节中的算法结合使用）是 Boltzmann 探索策略，或者称为 softmax 探索策略。动作选择策略（action selection strategy）仍然是随机的，但选择的概率通过相对 Q 值进行加权。这使得学习器更有可能选择非常好的动作，而几乎有相同的概率选中具有类似 Q 值的两个操作。这种方法的一般形式是：

$$P(a) = \frac{e^{\frac{Q(s,a)}{T}}}{\sum_i e^{\frac{Q(s,a_i)}{T}}} \quad (1.17)$$

```

for each episode do
   $s \in S$  is initialized as the starting state
   $t := 0$ 
  repeat
    choose an action  $a \in A(s)$ 
    perform action  $a$ 
    observe the new state  $s'$  and received reward  $r$ 
    update  $\tilde{T}$ ,  $\tilde{R}$ ,  $\tilde{Q}$  and/or  $\tilde{V}$ 
    using the experience  $\langle s, a, r, s' \rangle$ 
     $s := s'$ 
  until  $s'$  is a goal state
  
```

算法 3 在线强化学习的通用模板

其中， $P(a)$ 是选择动作 a 的概率， T 是温度参数（temperature parameter）。 T 的值越大，则选择策略越偏向纯粹的随机策略，而反之将偏向完全的贪婪策略。结合 ε -贪婪策略和 Boltzmann 探索策略可以用概率 $1-\varepsilon$ 来计算最优行为，否则根据公式 (1.17) 来计算动作 [Wiering, 1999]。

另一个促进探索的简单方法是乐观的 Q 值初始化；在学习开始的时候，可以将所有的 Q 值初始化为高值，例如事先确定的上界。因为 Q 值在学习过程中会减少，没有经过多次尝试的动作在使用时会有足够大的值，例如使用 Boltzmann 探索策略。另一个具有类似效果的解决方案是将计数器的值与一个特定状态 - 动作对（state-action pair）被选中的次数保持一致。

1.7.1 时序差分学习

时序差分（Temporal Difference, TD）学习算法在其他估计值的基础上学习。世界上的每一个步骤都产生一个学习样例，这些样例可以根据即时回报和下一个状态或状态 - 动作对的估计值来带来一些价值。

一个直观的例子描述如下 [Sutton and Barto, 1998]。想象一下，你必须预测客人什么时候能到达你家的小餐馆。在烹饪之前，根据那个订单，你必须去超市、肉店和酒庄。你有所有位置之间驾驶时间的估计值，你认为你能设法在最后 10 分钟内光顾最后两家店，但由于拥堵，你估计去超市需要一个半小时。基于这样的预测，你必须通知你的客人可在 18:00 之后到达。一旦你到了超市，只需要 10 分钟的时间购买所有你需要的东西，你可以调整你的时间：估计至多 20 分钟可回到家。然而，从肉店到酒庄，你会看到沿途交通拥堵，这条路需要你至少 30 分钟才能到达。最后，你比首次预测的晚 10 分钟到达。这个例子的底线是，每当你获得了两个步骤间的新的信息后，你可以调整何时到家的估值。根据你花在路上



的实际时间，每一次你可以调整你的估计。这是 TD 学习的主要原则：你不必等到试验结束再沿着你的路径进行更新。

TD 方法基于其他值的估计学习值的估计，即引导 (bootstrapping)。这些方法对动态规划优势是：这些方法不需要一个马尔可夫决策过程模型。另一个优点是，这些方法可以很自然地实现一个在线的、增量的学习方式，使得这些方法可以很容易地在不同的情况下使用。只要经验丰富的路径值得到更新并起作用，扫描完整的状态空间不是必要的。

1. TD(0)

TD(0) 是 TD 学习算法中的一员 [Sutton, 1988]。TD(0) 解决了预测问题，也就是说，TD(0) 以一种在线的增量的方式来估算策略。TD(0) 可用于评估策略，并通过以下更新规则^①进行工作：

$$V_k(s) = V_k(s) + \alpha(r + \gamma V_k(s') - V_k(s))$$

其中， $\alpha \in [0,1]$ 是学习率，学习率决定多少值得到更新。

这个备份是在经历从状态 s 到 s' 的转换过程中执行的，同时接收到奖励 r 。与公式 (1.4) 中使用的动态规划备份的不同之处在于 TD(0) 使用的更新仍然是通过使用 bootstrapping 算法进行的。但是，TD(0) 是基于一个观察到的转换，即使用一个采样的备份而不是一个完整的备份。只使用一个后续状态的值，而不是所有可能的后续状态的加权平均值。当使用价值函数 V^* 做动作选择时，需要一个模型计算所有动作结果的期望值（例如，见公式 (1.4)）。

为了学习收敛，学习率 α 要适当降低。有时学习率可以分别定义为在一个状态中的状态 $\alpha(s)$ ，在这种情况下，它可以依赖于状态被访问的次数。接下来的两个算法直接从样本中学习 Q 函数，不需要动作选择转型模式。

30

Require: discount factor γ , learning parameter α
 initialize Q arbitrarily (e.g. $Q(s,a) = 0, \forall s \in S, \forall a \in A$)
for each episode do
 s is initialized as the starting state
 repeat
 choose an action $a \in A(s)$ based on Q and an exploration strategy
 perform action a
 observe the new state s' and received reward r
 $Q(s,a) := Q(s,a) + \alpha \left(r + \gamma \cdot \max_{a' \in A(s')} Q(s',a') - Q(s,a) \right)$
 $s := s'$
 until s' is a goal state

算法 4 Q 学习，详见 [Watkins and Dayan, 1992]

2. Q 学习

在算法 4 中，在无模型的情况下估计 Q 值函数的最基本和最流行的方法之一是 Q 学习算法，见 [Watkins, 1989; Watkins and Dayan, 1992]。

Q 学习的基本思想是基于反馈（即奖励）和学习器的 Q 价值函数增量估计 Q 值的动作。更新规则是 TD 学习的升级版，TD 学习使用 Q 值和内置的 max 运算符 (max-operator)，而不是下一个状态的 Q 值：

① 学习参数 α 应符合其值的一些标准和变化的方式。在这一部分的算法中，通常选择一个小的固定的学习参数，或者每次迭代都递减。



$$Q_{k+1}(s_t, a_t) = Q_k(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q_k(s_{t+1}, a) - Q_k(s_t, a_t) \right) \quad (1.18)$$

从状态 s_t 到 s_{t+1} , 学习器在环境中在接收奖励 r_t 时使用动作 a_t 进行操作。在状态 s_t 上执行的动作 a_t 的 Q 值上更新。

Q 学习是探索不敏感的。这意味着 Q 学习将收敛于最优策略, 而不考虑所遵循的探索策略, 假设每个状态-动作对都被访问了无数次, 而学习参数 α 被适当地减少了 [Watkins and Dayan, 1992; Bertsekas and Tsitsiklis 1996]。

3. SARSA 学习

Q 学习是一种偏离策略的学习算法, 这意味着在遵循一些探索策略 π 的同时, Q 学习的目标是估计最优策略 π^* 。一个相关的策略算法是, 它学习了学习器实际执行的策略的 Q - 价值函数, 即状态-动作-奖励-状态-动作 (State-Action-Reward-State-Action, SARSA) 算法 [Rummery and Niranjan, 1994; Rummery, 1995; Sutton, 1996]。SARSA 学习使用以下更新规则:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha \left(r_t + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \right) \quad (1.19)$$

其中, 动作 a_{t+1} 是由当前策略执行的状态为 s_{t+1} 的动作。注意, Q 学习中的 \max 运算符被根据策略所估计的下一个动作的值所取代。最优价值函数的极限条件下所有状态和动作都是无限次的, 这个学习算法仍然会收敛, 并且策略收敛于贪婪策略 [Singh et al, 2000]。SARSA 在非平稳环境中是特别有用的。在这种情况下, 永远不会达到一个最优策略。如果使用函数逼近 (function approximation), SARSA 也很有用。因为在使用时, 无策略方法 (off-policy method) 可能会产生分歧。

4. 演员-评论家学习

先于 Q 学习和 SARSA 的算法是演员-评论家学习 (actor-critic learning), 见 [Witten, 1977; Barto et al, 1983; Konda and Tsitsiklis, 2003]。这个 TD 方法的分支保持与价值函数无关的一个独立的策略。这一策略称为演员而价值函数 (评论家)。评论家通常是一个状态-价值函数, 评估或评论演员 (actor) 所执行的动作的好坏。选择动作后, 评论家使用 TD 误差评估动作:

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

这个误差的目的是加强或削弱这个状态中的动作的选择。在某些状态 s 中的动作 a 的偏好可以表示为 $p(s, a)$, 这样的偏好可以使用下式修改:

$$p(s_t, a_t) := p(s_t, a_t) + \beta \delta_t$$

其中, 参数 β 的大小决定了更新的大小。有其他版本的演员-评论家的方法, 主要的不同在于偏好是如何改变的, 或经验是如何使用的。例如, 使用资格路径 (eligibility trace), 详见下一节。有一个单独的策略表示的优点是: 如果有很多动作, 或者当动作空间是连续的, 那么就不需要考虑所有的操作的 Q 值, 来选择其中的一个。第二个优点是, 它们可以自然地学习随机策略 (stochastic policies)。此外, 可以使用关于策略约束的先验知识, 见 [Framling, 2005]。

5. 平均奖励时序差分学习

我们已经在贴现的、无限时域马尔可夫决策过程的情况下讲解了 Q 学习和相关的算法。

[32] Q 学习也可以适用于平均奖励框架, 例如在 R 学习算法中 [Schwartz, 1993], 对平均奖励框



架的其他算法的扩展详见 [Mahadevan, 1996]。

1.7.2 蒙特卡罗方法

其他使用更公正估计的算法是蒙特卡罗 (Monte Carlo, MC) 技术。蒙特卡罗技术保持对状态-动作对和未来的奖励 (回报) 的频率计数, 并根据这些估计建立它们的值。蒙特卡罗技术只需要基于样本来估计平均样本的回报。例如, 在蒙特卡罗策略的评估中, 对于每个状态 $s \in S$, 保留了所有从 s 获得的状态, 一个状态 $s \in S$ 的值是它们的平均值。换言之, 蒙特卡罗技术将长期奖励作为一个随机变量, 并以其估计的采样平均值作为一个随机变量。

与一步 TD 方法相比, 蒙特卡罗技术基于计算平均样本回报过程中观察到的相互作用来估计数值。特别是对于周期性任务, 因为从完全回报中可以得到样本, 蒙特卡罗技术是非常有用的。使用蒙特卡罗技术的一种方法是: 使用在策略迭代的评估步骤中使用。然而, 由于采样依赖于当前策略 π , 策略 π 只评估所建议的动作用的回报。因此, 探索在这里是至关重要的, 就像其他无模型的方法一样。

可以区分每次访问蒙特卡罗技术之间的区别, 蒙特卡罗技术在每个阶段对所有状态 $s \in S$ 的访问计算平均值, 第一次访问蒙特卡罗技术第一次访问的状态 $s \in S$ 所获得的回报的平均值。对于当前策略 π , 随着时间的推移, 两种改进版本都会收敛到 V^π 。蒙特卡罗技术也可以应用于估计动作值的问题。确保足够的探索的一种方法是使用探索启动 (exploring start), 即每个状态-动作对具有一个非零的概率被选为初始对 (initial pair)。蒙特卡罗技术可以用于策略控制和非策略控制, 一般的模式符合广义策略迭代过程。蒙特卡罗技术没有引导, 这使得蒙特卡罗技术不那么依赖于马尔可夫假设。尽管 TD 方法使用引导, 但其过于注重采样的经验。

1. 学习一个模型

我们已经描述了几种学习价值函数的方法。间接的强化学习或基于模型的强化学习也可以用于评估一个基础 MDP 的模型。在交互过程中所经历的样本转换概率 (sample transition probability) 的平均值可以用来逐步估计转换概率。同样可用于即时奖励 (immediate reward)。间接的强化学习算法利用这种方法在基于模型的学习和无模型的学习之间取得平衡。它们本质上是无模型的, 但是使用无模型的强化学习的同时学习转换和奖励模型 (reward model), 并使用该模型进行更有效的价值函数学习 (value function learning), 详见下一节。这方面的一个例子是 DYNA 模型 [Sutton, 1991a]。通常采用模型学习的另一种方法是优先扫描 (prioritized sweeping), 见 [Moore and Atkeson, 1993]。学习模型对于在连续的空间中学习也是非常有用的, 在这些空间中, 转换模型被定义为底层 (无限) 状态空间的离散化版本 [Großmann, 2000]。

2. 与动态规划的关系

本节中的方法基本上解决了与动态规划技术类似的问题。强化学习的方法可以看作异步的动态规划。虽然这两种方法有一些重要的差异。

强化学习方法通过以下方式避免来自于动态规划的穷尽扫描 (exhaustive sweep): 限制计算或近邻扫描, 采样在附近的轨迹, 无论是真实的或模拟的。这可以利用许多状态在实际轨迹 (actual trajectory) 中发生概率较低的情况。在动态规划中使用的备份通过使用采样 (sampling) 来简化。而不是生成和评估一个状态所有可能的直接继任者 (possible immediate successors), 对备份影响的评估是从适当的分布中进行采样的。蒙特卡罗技术使用上述方

33



法来完全根据回报样本进行评估，没有引导使用其他值、采样值和状态值。此外，重点研究强化学习中的学习（动作）价值函数，是适用于函数逼近的方法。通过使用数值回归算法（numeric regression algorithm）来表示价值函数和策略比用查找表表示更简洁，而不会破坏标准的强化学习的交互过程；可以将更新值输入到回归引擎（regression engine）中。

有趣的一点是，一方面 Q 学习与值迭代类似，另一方面 SARSA 与策略迭代类似。在前两种方法中，更新立即将策略评估和策略改进结合为使用 \max 运算符的步骤。与之相反，后两种方法分别对策略进行评估和完善。在这方面，值迭代可以直接视为无策略的，因为值迭代的目的是估计 V^* ，而策略迭代估计当前策略的值且是有策略的。然而，在基于模型的设置中，区别仅仅是表面的，因为使用模型，而不是受策略分配影响的样本，就可以知道状态和奖励的分布情况。

1.7.3 高效的探索和价值更新

在上一节中的方法表明，可以从与环境交互的样本中学习预测和控制，而不需要获得一个马尔可夫决策过程模型。这些方法的一个问题是，它们往往需要大量的经验来收敛。在这一节中，我们描述一些用于加快学习的扩展。改进的一个方向在于探索。原则上来说，可以使用模型评估（model estimation），直到知道马尔可夫决策过程的所有内容，但这太耗时了。使用更多的信息，使用更为集中的探索程序，用来更有效地产生经验。另一个方向是在每个步骤中使用经验来更新价值函数的多个值。改进的探索产生更好的样本，而改进的更新将会从样本中挖掘出更多的信息。

1. 高效率的探索

我们已经遇到 ϵ -贪婪策略和 Boltzmann 探索策略。虽然常用，这些都是比较简单的无向探索方法。这些方法主要是由随机性驱动的。此外，这些方法是无状态的，即在不知道迄今为止已经探索了哪些领域的状态空间的情况下就启动探索。

在文献中提出了一大类使用关于学习过程的额外信息的定向方法。这些方法的重点是对状态空间进行更统一的探索，并平衡发现新信息相对于利用现有知识的相对收益。大多数方法都使用或学习与强化学习并行的马尔可夫决策过程模型。此外它们还学习了一个探索价值函数。

有定向探索（directed exploration）的几个选项是可用的。方法间的一个区别是，在决定探索时，是否要在本地（例如，探索单个状态-动作对）或全局范围内考虑有关部分或完整的状态空间的信息。此外，还有其他几类探索算法。

基于计数器的方法（counter-based method）或基于近因的方法（recency-based method）记录了状态-动作对的访问频率，或者多久之前访问了状态-动作对。基于误差的方法使用基于状态值错误的探索红利的方法（exploration bonus）。其他方法探索一个状态值的不确定性，或对状态的当前值的可信度。它们决定是否通过计算探索动作获得比预期更大的奖励的概率来探索。区间估计（interval estimation, IE）方法是这种方法的一个例子，见 [Kaelbling, 1993]。区间估计使用一个统计模型来测量每个 $Q(s,a)$ 值的不确定性程度。可以根据每个 Q 值的可能值计算上界，并执行具有最高上界的动作。如果所采取的动作是一个糟糕的选择，那么当统计模型更新时，上界将会减少。良好的动作将继续有较高的上界，并将经常被选择。与基于计数器的探索和基于近因的探索相反，区间估计关注的是动作探索，而不是状态空间探索。Wiering 在基于模型的区间估计算法中引入了基于模型的强化学习的扩展



[Wiering, 1999; Wiering and Schmidhuber, 1998a], 该算法用于对转换概率的估计。

另一种显式处理探索-运用平衡的方法是 E^3 方法 [Kearns and Singh, 1998]。 E^3 代表显式的探索与运用。 E^3 通过收集统计数据更新环境模型来学习。状态空间分为已知的部分和未知的部分。每一步都有一个决定, 即已知的部分是否有足够的机会获得奖励, 或者是否应该探索未知的部分以获得更多的奖励。该算法的一个重要方面是它是在计算时间上具有可证明边界且接近最优的(表格式的)强化学习算法。该方法被扩展为更一般的算法 R-MAX, 见 [Brafman and Tennenholtz, 2002]。该方法也提供了一个多项式限定的计算时间以达到接近最优策略。最后一个例子采用了一种基于马尔可夫决策过程更复杂特性的高效的定向探索方法 [Ratitch, 2005], 比如对状态转换的熵度量 (entropy measure)。这种方法的一个有趣的特点是, 这些特征可以在学习之前计算出来, 并与其他探索方法结合使用, 从而提高这些方法的动作的质量。关于探索策略更详细的报道, 可以参考 [Ratitch, 2005; Wiering, 1999]。

35

2. 指导和变形

探索方法可以用来加速学习, 把注意力集中在探索状态空间的相关领域。探索方法主要使用在学习之前或在学习过程中的统计数据。然而, 有时有更多的信息可以用来指导学习器。例如, 如果一个领域的合理策略可用, 这个领域就可以用来生成比(随机)探索更有用的学习样本。实际上, 在指定最优策略方面, 人类通常表现得非常糟糕, 但在指定合理的策略方面却相当不错[⊖]。

行为克隆 (behavioral cloning) 的工作在指导光谱 (guidance spectrum) 上有一个极端的观点, 即目标是模仿专家跟踪的例子动作 [Bain and Sammut, 1995]。这种类型的指导使学习更多的朝着监督学习的方向发展。另一种方法是变形 (shaping), 见 [Mataric, 1994; Dorigo and Colombetti, 1997; Ng et al, 1999]。变形使奖励更接近动作的子目标, 从而鼓励学习器通过更有效地搜索策略空间来提高动作的质量。这也涉及给予适当的子目标奖励的一般问题, 以及任务难度的逐渐增加。学习器可以接受越来越多困难问题的训练, 这也可以认为是一种指导的形式。

还有其他各种机制可以用来为强化学习算法提供指导, 如分解 [Dixon et al, 2000], 为更好探索的启发式规则 [Framling, 2005], 以及与其他相关问题的各种类型的知识转换, 如 [Konidaris, 2006]。

36

3. 资格迹

在蒙特卡罗方法中, 更新是基于观察到的奖励的整个序列的, 直到一个周期结束。在 TD 方法中, 这些估计基于直接奖励和下一个状态的样本。中间方法是使用 n -步进-返回 (n -step-truncated-return) $R_t^{(n)}$ 从一系列的回报中得到:

$$R_t^{(n)} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^n V_t(s_{t+n})$$

有了上式, 我们就可以根据几个 n -步来计算值的更新。TD 的系列 $TD(\lambda)$, 满足 $0 \leq \lambda \leq 1$ 条件, 结合 n -步回报, 相应的加权比例为 λ^{n-1} 。

问题是, 我们必须无限期地等待计算 $R_t^{(\infty)}$ 。这一观点有助于对 n -步备份进行理论分析和理解。这一观点称为 $TD(\lambda)$ 算法的前视 (forward view)。然而, 要实现这种更新的常用方法称为 $TD(\lambda)$ 算法的后视 (backward view) 且使用资格迹, 这是 $TD(\lambda)$ 算法前视的增量

⊖ 引自 Leslie Kaelbling 在强化学习欧洲研讨会 (European Workshop on Reinforcement Learning, EWRL) 的特邀报告。

实现。

资格迹是一种优雅地方式执行 n -步备份的方式。对于每一个状态 $s \in S$ 来说, 资格 $e_t(s)$ 是保存在内存中的。它们都初始化为 0, 并根据以下公式递增:

$$e_t(s) = \begin{cases} \gamma \lambda e_{t-1}(s) & , s \neq s_t \\ \gamma \lambda e_{t-1}(s) + 1 & , s = s_t \end{cases}$$

其中, λ 是路径衰减参数 (trace decay parameter)。每次访问状态时, 每个状态的路径都会增加, 否则会以指数形式递减。现在, δ_t 是在阶段 t 的时间误差 (temporal difference error):

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

在每一步中, 所有的状态都按自己的资格迹进行更新:

$$V(s) := V(s) + \alpha \delta_t e_t(s)$$

可以证明资格迹的前视和后视是等价的 [Sutton and Barto, 1998]。对于 $\lambda=1$ 来说, TD(λ) 基本上与蒙特卡罗技术是相同的, 因为考虑了完整回报。对于 $\lambda=0$ 来说, TD(λ) 只使用所有一步强化学习算法的即时回报。资格迹是从 n -步回报中学习的一般机制。这些方法可以与我们在上一节中描述的所有无模型方法相结合。在 Q(λ) 算法中, Watkins 结合了 Q 学习和资格迹算法 [Watkins, 1989]。[Peng and Williams, 1996] 提出了一个类似的算法, Wiering 等提出了高效版本的 Q(λ), 见 [Wiering and Schmidhuber, 1998b; Reynolds, 2002]。将资格迹与学习控制 (learning control) 相结合的问题是: 必须特别注意在探索动作的情况, 能打破当前策略 n -步回报的预期意义。在 Watkins 开发的版本中, 每当采取一个探索性的动作时重置资格迹 [Watkins, 1989]。[Peng and Williams, 1996] 版本在这方面更有效, 因为不是每一次都将路径设为零。在这方面, 因为动作选择是有策略的, SARSA(λ) 更加安全 [Sutton and Barto, 1998]。另一个策略学习算法是 QV(λ), 见 [Wiering, 2005]。在 QV(λ) 学习中, 学习两个价值函数; TD(λ) 用于学习一个状态函数 V , 且一步 Q 学习算法用于学习的基于 V 的状态-动作价值函数。

4. 学习和使用模型: 学习与规划

虽然强化学习方法没有马尔可夫决策过程模型也能运行, 有个模型可以加快学习和偏置搜索 (bias exploration)。学习模型也有助于做更有效的价值更新。一个总的方针是: 当经验是昂贵的, 相应的回报是学习模型。强化学习中的模型学习通常针对马尔可夫决策过程定义的特定的学习任务, 即由奖励和目标确定的任务。一般来说, 学习一个模型通常是有用的, 因为它提供了动态的环境知识, 并可以用于其他任务, 详见 [Drescher, 1991]。

DYNA 架构是一个用模型来放大经验的简单方式 [Sutton, 1990, 1991b,a; Sutton and Barto, 1998]。DYNA-Q 实现见算法 5, 其结合了 Q 学习与规划。在连续循环中, 通过一个不断更新的模型, Q 学习算法将进行一系列额外的更新。DYNA 与环境交互的需求较小, 因为 DYNA 重放经验以执行更多的价值更新。

使用学习模型的更多使用经验的相关方法是优先扫描 (Prioritized Sweeping, PS) [Moore and Atkeson, 1993]。在 DYNA 中, 优先扫描优先考虑基于价值的变化更新, 而不是选择要随机更新的状态。一旦一个状态被更新, 通过检查转换模型的信息, 并检查这些状态是否也将被更新, 优先扫描将考虑所有的能到达这个状态的状态。更新的顺序是由价值更新的大小决定的。一般机制可以归纳如下。在每一步, 1) 从当前状态中存储旧的值。2) 一个使用学习模型 (learned model) 对状态值进行完整的备份。3) 将当前状态的优先级设置为 0。

4) 计算数值的变化 δ , 并作为备份的结果。5) 使用这种差异来修改当前状态的前身 (由模型决定)。

```

Require: initialize  $Q$  and Model arbitrarily
repeat
   $s \in S$  is the start state
   $a := \varepsilon\text{-greedy}(s, Q)$ 
  update  $Q$ 
  update Model
  for  $i := 1$  to  $n$  do
     $s :=$  randomly selected observed state
     $a :=$  random, previously selected action from  $s$ 
    update  $Q$  using the model
until Sufficient Performance
  
```

算法 5 DYNA-Q 算法 [Sutton and Barto, 1998]

38

所有导致当前状态的状态 $\delta \times T$ 都得到优先更新, 其中 T 是一个继承状态 (successor state) 导致当前状态更新的概率。值备份的数量是在算法中设置的一个参数。总体而言, 优先扫描将备份集中到预计最快减少错误的地方。在基于模型的强化学习中使用规划的另一个例子见 [Wiering, 2002]。

1.8 总结

本章为马尔可夫决策过程、动态规划和强化学习提供了必要的背景知识。将在以后的章节中讨论的许多算法的核心要素是贝尔曼方程、值更新、探索和采样。

参考文献

- Bain, M., Sammut, C.: A framework for behavioral cloning. In: Muggleton, S.H., Furakawa, K., Michie, D. (eds.) Machine Intelligence, vol. 15, pp. 103–129. Oxford University Press (1995)
- Barto, A.G., Sutton, R.S., Anderson, C.W.: Neuronlike elements that can solve difficult learning control problems. IEEE Transactions on Systems, Man, and Cybernetics 13, 835–846 (1983)
- Barto, A.G., Bradtke, S.J., Singh, S.: Learning to act using real-time dynamic programming. Artificial Intelligence 72(1), 81–138 (1995)
- Bellman, R.E.: Dynamic Programming. Princeton University Press, Princeton (1957)
- Bertsekas, D.P.: Dynamic Programming and Optimal Control, vol. 1, 2. Athena Scientific, Belmont (1995)
- Bertsekas, D.P., Tsitsiklis, J.: Neuro-Dynamic Programming. Athena Scientific, Belmont (1996)
- Bonet, B., Geffner, H.: Faster heuristic search algorithms for planning with uncertainty and full feedback. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 1233–1238 (2003a)
- Bonet, B., Geffner, H.: Labeled RTDP: Improving the convergence of real-time dynamic programming. In: Proceedings of the International Conference on Artificial Intelligence Planning Systems (ICAPS), pp. 12–21 (2003b)
- Boutilier, C.: Knowledge Representation for Stochastic Decision Processes. In: Veloso, M.M., Wooldridge, M.J. (eds.) Artificial Intelligence Today. LNCS (LNAI), vol. 1600, pp. 111–152. Springer, Heidelberg (1999)

39

- Boutilier, C., Dean, T., Hanks, S.: Decision theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research* 11, 1–94 (1999)
- Brafman, R.I., Tennenholtz, M.: R-MAX - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research (JMLR)* 3, 213–231 (2002)
- Dean, T., Kaelbling, L.P., Kirman, J., Nicholson, A.: Planning under time constraints in stochastic domains. *Artificial Intelligence* 76, 35–74 (1995)
- Dixon, K.R., Malak, M.J., Khosla, P.K.: Incorporating prior knowledge and previously learned information into reinforcement learning agents. Tech. rep., Institute for Complex Engineered Systems, Carnegie Mellon University (2000)
- Dorigo, M., Colombetti, M.: *Robot Shaping: An Experiment in Behavior Engineering*. The MIT Press, Cambridge (1997)
- Drescher, G.: *Made-Up Minds: A Constructivist Approach to Artificial Intelligence*. The MIT Press, Cambridge (1991)
- Ferguson, D., Stentz, A.: Focussed dynamic programming: Extensive comparative results. Tech. Rep. CMU-RI-TR-04-13, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania (2004)
- Främling, K.: Bi-memory model for guiding exploration by pre-existing knowledge. In: Driessens, K., Fern, A., van Otterlo, M. (eds.) *Proceedings of the ICML-2005 Workshop on Rich Representations for Reinforcement Learning*, pp. 21–26 (2005)
- Großmann, A.: Adaptive state-space quantisation and multi-task reinforcement learning using constructive neural networks. In: *From Animals to Animats: Proceedings of The International Conference on Simulation of Adaptive Behavior (SAB)*, pp. 160–169 (2000)
- Hansen, E.A., Zilberstein, S.: LAO*: A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence* 129, 35–62 (2001)
- Howard, R.A.: *Dynamic Programming and Markov Processes*. The MIT Press, Cambridge (1960)
- Kaelbling, L.P.: *Learning in Embedded Systems*. The MIT Press, Cambridge (1993)
- Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4, 237–285 (1996)
- Kearns, M., Singh, S.: Near-optimal reinforcement learning in polynomial time. In: *Proceedings of the International Conference on Machine Learning (ICML)* (1998)
- Koenig, S., Liu, Y.: The interaction of representations and planning objectives for decision-theoretic planning. *Journal of Experimental and Theoretical Artificial Intelligence* 14(4), 303–326 (2002)
- Konda, V., Tsitsiklis, J.: Actor-critic algorithms. *SIAM Journal on Control and Optimization* 42(4), 1143–1166 (2003)
- Konidaris, G.: A framework for transfer in reinforcement learning. In: *ICML-2006 Workshop on Structural Knowledge Transfer for Machine Learning* (2006)
- Kushmerick, N., Hanks, S., Weld, D.S.: An algorithm for probabilistic planning. *Artificial Intelligence* 76(1-2), 239–286 (1995)
- Littman, M.L., Dean, T., Kaelbling, L.P.: On the complexity of solving Markov decision problems. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 394–402 (1995)
- Mahadevan, S.: Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning* 22, 159–195 (1996)
- Maloof, M.A.: Incremental rule learning with partial instance memory for changing concepts. In: *Proceedings of the International Joint Conference on Neural Networks*, pp. 2764–2769 (2003)
- Mataric, M.J.: Reward functions for accelerated learning. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 181–189 (1994)
- Matthews, W.H.: *Mazes and Labyrinths: A General Account of their History and Developments*. Longmans, Green and Co., London (1922); *Mazes & Labyrinths: Their History & Development*. Dover Publications, New York (reprinted in 1970)
- McMahan, H.B., Likhachev, M., Gordon, G.J.: Bounded real-time dynamic programming:

- RTDP with monotone upper bounds and performance guarantees. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 569–576 (2005)
- Moore, A.W., Atkeson, C.G.: Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning* 13(1), 103–130 (1993)
- Ng, A.Y., Harada, D., Russell, S.J.: Policy invariance under reward transformations: Theory and application to reward shaping. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 278–287 (1999)
- Peng, J., Williams, R.J.: Incremental multi-step Q-learning. *Machine Learning* 22, 283–290 (1996)
- Puterman, M.L.: *Markov Decision Processes—Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York (1994)
- Puterman, M.L., Shin, M.C.: Modified policy iteration algorithms for discounted Markov decision processes. *Management Science* 24, 1127–1137 (1978)
- Ratitch, B.: On characteristics of Markov decision processes and reinforcement learning in large domains. PhD thesis, The School of Computer Science, McGill University, Montreal (2005)
- Reynolds, S.I.: Reinforcement learning with exploration. PhD thesis, The School of Computer Science, The University of Birmingham, UK (2002)
- Rummery, G.A.: Problem solving with reinforcement learning. PhD thesis, Cambridge University, Engineering Department, Cambridge, England (1995)
- Rummery, G.A., Niranjan, M.: On-line Q-Learning using connectionist systems. Tech. Rep. CUED/F-INFENG/TR 166, Cambridge University, Engineering Department (1994)
- Russell, S.J., Norvig, P.: *Artificial Intelligence: a Modern Approach*, 2nd edn. Prentice Hall, New Jersey (2003)
- Schaeffer, J., Platt, A.: Kasparov versus deep blue: The re-match. *International Computer Chess Association Journal* 20(2), 95–101 (1997)
- Schwartz, A.: A reinforcement learning method for maximizing undiscounted rewards. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 298–305 (1993)
- Singh, S., Jaakkola, T., Littman, M., Szepesvari, C.: Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning* 38(3), 287–308 (2000)
- Sutton, R.S.: Learning to predict by the methods of temporal differences. *Machine Learning* 3, 9–44 (1988)
- Sutton, R.S.: Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 216–224 (1990)
- Sutton, R.S.: DYNA, an integrated architecture for learning, planning and reacting. In: Working Notes of the AAAI Spring Symposium on Integrated Intelligent Architectures, pp. 151–155 (1991a)
- Sutton, R.S.: Reinforcement learning architectures for animats. In: From Animals to Animats: Proceedings of The International Conference on Simulation of Adaptive Behavior (SAB), pp. 288–296 (1991b)
- Sutton, R.S.: Generalization in reinforcement learning: Successful examples using sparse coarse coding. In: Proceedings of the Neural Information Processing Conference (NIPS), pp. 1038–1044 (1996)
- Sutton, R.S., Barto, A.G.: *Reinforcement Learning: an Introduction*. The MIT Press, Cambridge (1998)
- Tash, J., Russell, S.J.: Control strategies for a stochastic planner. In: Proceedings of the National Conference on Artificial Intelligence (AAAI), pp. 1079–1085 (1994)
- Watkins, C.J.C.H.: Learning from delayed rewards. PhD thesis, King's College, Cambridge, England (1989)
- Watkins, C.J.C.H., Dayan, P.: Q-learning. *Machine Learning* 8(3/4) (1992); Special Issue on Reinforcement Learning
- Wiering, M.A.: Explorations in efficient reinforcement learning. PhD thesis, Faculteit der Wiskunde, Informatica, Natuurkunde en Sterrenkunde, Universiteit van Amsterdam

(1999)

Wiering, M.A.: Model-based reinforcement learning in dynamic environments. Tech. Rep. UU-CS-2002-029, Institute of Information and Computing Sciences, University of Utrecht, The Netherlands (2002)

Wiering, M.A.: QV(λ)-Learning: A new on-policy reinforcement learning algorithm. In: Proceedings of the 7th European Workshop on Reinforcement Learning (2005)

Wiering, M.A., Schmidhuber, J.H.: Efficient model-based exploration. In: From Animals to Animats: Proceedings of The International Conference on Simulation of Adaptive Behavior (SAB), pp. 223–228 (1998a)

Wiering, M.A., Schmidhuber, J.H.: Fast online Q(λ). Machine Learning 33(1), 105–115 (1998b)

Winston, W.L.: Operations Research Applications and Algorithms, 2nd edn. Thomson Information/Publishing Group, Boston (1991)

Witten, I.H.: An adaptive optimal controller for discrete-time markov environments. Information and Control 34, 286–295 (1977)

高效的解决方案框架

批处理强化学习

Sascha Lange, Thomas Gabel, Martin Riedmiller

摘要

批处理强化学习 (Batch Reinforcement Learning, BRL) 是基于动态规划的强化学习的一个分支。批处理算法 (batch algorithm) 的最初定义是: 从一组固定的已知的先验转换样本中学习最佳可能策略的任务 (task)。在这个领域中开发出来的批处理算法能够很好地适应经典的在线例子: 学习器 (agent) 与环境 (environment) 不断交互 (interaction) 进行学习。由于收集到的数据的有效利用和学习过程的稳定性, 这一研究领域最近引起了大量的关注。在本章中, 我们介绍批处理强化学习的基本原理和理论, 描述最重要的算法, 包括基于核的近似动态规划 (Kernel-Based Approximate Dynamic Programming, KADP)、拟合 Q 迭代 (Fitted Q Iteration, FQI)、基于最小二乘策略的循环 (Least-Squares Policy Iteration, LSPI)、最小二乘时序差分 (Least-Squares Temporal Difference, LSTD) 学习算法、深度拟合 Q 迭代 (Deep Fitted Q Iteration, DFQ)、神经拟合 Q 迭代 (Neural Fitted Q Iteration, NFQ) 算法等, 并举例讨论在这个领域正在进行的研究, 简要地综述了批处理强化学习现实应用的例子。

2.1 简介

批处理强化学习是基于动态规划的强化学习。在过去的几年中, 批处理强化学习变得越来越重要。从历史上看, 批处理强化学习这个术语用来描述强化学习的设置。其中, 学习经验通常是从学习系统取样的转换 (transition) 组成的集合, 而所有的学习经验是固定的且拥有先验概率的 [Ernst et al, 2005a]。学习系统的任务是从给定的批量样本中推导出一个解决方案, 而这个解决方案通常是一个最优策略。

在本章中, 我们将放宽具有先验知识的固定训练集这一假设。批处理模式算法的关键好处在于: 这些算法处理一批转换并从中得到最好的一个转换, 而这个集合不是固定的。从这个角度看, 批处理强化学习算法的特点包含两个基本的组成成分: 所有观察到的转换在整个转换中同时的存储和更新 (update), 即拟合 (fitting)。特别是允许定义“增长批处理” (growing batch) 的方法, 这个方法允许扩展样本经验的集合以逐步改进算法的性能。从交互的角度来看, 增长批处理方法最大程度地减少了批处理方法和纯粹的在线学习方法之间的差异。

批处理的好处是: 批处理算法在学习过程中具有良好的稳定性和较高的数据效率, 这些好处使研究者和开发人员对批处理算法有极大兴趣。学习算法 (如 Q 学习) 通常需要与环境大量交互, 直到收敛 (convergence) 于好的策略。因为这个特性, 学习算法难于直接运用于实际的问题。许多与批处理强化学习相近的方法往往具有更快的收敛速度。很多将批处理强化学习成功应用于现实系统的文献已经发表, 参见 2.6.2 节和 2.6.5 节。

在这一章中, 我们将首先定义批处理强化学习的主要研究问题和相关的衍生方式, 从而

形成批处理强化学习的问题空间。我们简短地回顾所有构成现代批处理强化学习算法基础的中心想法的发展历程。通过对关键科学问题的定义和引入，我们将阐述批处理强化学习包含的最重要的算法。我们将讨论批处理强化学习的理论特性，以及一些与实际应用高度相关的衍生算法。因为神经拟合 Q 迭代是在真实系统中学习的一个强大的工具，所以批处理强化学习的衍生算法还包括神经拟合 Q 迭代和一些相关的应用。随着批处理强化学习成功应用于控制策略的视觉学习和解决分布式调度问题等领域，我们将简单讨论正在进行的研究。

2.2 批处理强化学习问题

过去，批处理强化学习被定义为一类解决特定学习问题的算法，即批处理强化学习问题。

2.2.1 批处理学习问题

[Sutton and Barto, 1998] 定义的批处理强化学习问题的通用学习框架中，批处理强化学习问题的核心任务是：在熟悉的学习器-环境的循环中，如何发现能够最大化奖励的总和的期望值的策略。然而，与一般情况不同的是，在批处理学习（batch learning）问题中，学习器本身是不允许在学习过程中与系统交互的。学习器的作用不是观察状态 s ，尝试一个动作 a 并根据接下来的状态 s' 和奖励 r 改变策略，学习器只接收一个从环境中取得的 p 个转换的样本 (s, a, r, s') ，形成集合 $\mathcal{F} = \{(s_t, a_t, r_{t+1}, s_{t+1}) | t = 1, \dots, p\}^\ominus$ 。

46

在批处理强化学习的最一般的情况下，学习器不能对转换的采样过程做任何假设。学习器可以采用任意甚至完全随机的采样策略，不一定都是从状态-动作（state-action）空间 $S \times A$ 中均匀采样，甚至不需要沿连接的轨迹（trace）采样。为了使用这些有价值的信息，学习器必须使用能够与环境交互的策略。在应用阶段，策略是固定的且没有因为有了新的意见而进一步改进。由于学习器本身是不允许与环境交互的，且给定集合中包含的转换的数量通常是有限的，因此学习器不可能总是给出最优策略。通常，强化学习的目标是通过学习得到最优策略。然后，因此学习器的目标已经演化到从给定数据中学习到最佳的可能策略。

学习的整个过程可以分为相互独立的三个步骤：探索环境，收集状态转换和奖励；学习一个策略；以及运用已经学到的策略。这三个步骤是按照时间顺序执行的，这三个步骤之间数据转换的细节在图 2.1 中进行了描述。显然，探索的困难不完全来自于批处理强化学习问题，而且探索也不是学习任务的部分。

47

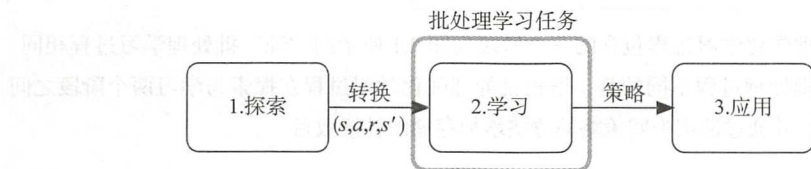


图 2.1 批处理强化学习的三个不同阶段：1）通过一个采样策略收集转换；2）运用强化学习或者批处理强化学习算法，从转换的集合中学习到最优策略；3）学习策略的应用。探索不是批处理学习任务的一个部分。在应用阶段，也不是学习任务的一部分，策略保持不变且没有进一步改进

\ominus 本章介绍的方法都假设从一个离散时间马尔可夫决策过程（见第 1 章）中采样的一个马尔可夫状态表示和转换 \mathcal{F} 。仅用于部分可观察决策过程的处理参见第 12 章。

2.2.2 增长批处理学习问题

过去,批处理强化学习的最主要问题是算法的开发,然而,现代的批处理强化学习很少在批处理学习问题中单独使用。在实践中,探索对于学到的策略的质量有重要影响。批处理算法包含的转换分布必须体现系统真正的转换概率(transition probability),从而推导出好的策略。实现这一点的最简单的方法就是从系统本身的训练样本中取样,只需要与系统进行简单的交互即可。但是,当从实际系统中采样时,另一个方面变得非常重要:学习过程中转换所覆盖的状态空间的大小。如果取样的样本没有涵盖重要的区域,例如与目标状态相接近的状态,从数据学习到一个好的策略明显是不可能的,因为学习所需要的重要的有价值的信息都缺失了。这是在实践中真正问题。一个完全“无知”的策略(比如一个完全随机的策略)往往没有能力实现对状态空间的全面覆盖,特别是在寻找合适的起始状态与到达期望的终止状态的时候。为了有能力探索不在起始状态附近但有价值的区域,需要对好的策略有一个粗略的想法。

这就是第三类强化学习问题的衍生算法成为通行方法的主要原因。这些算法定位于单纯的在线学习问题和单纯的批处理问题之间,因为这第三类学习问题的主要想法在探索阶段和学习阶段之间来回交换。其中,在探索阶段,一组训练样本来自于与系统的交互;在学习阶段,图 2.2 中的所有观察样本都用到了,称为增长批处理学习问题。在文献中,增长批处理方法可以通过几种不同的形式出现;在探索阶段和学习阶段之间来回交换的次数可以从两次到最大次数之间选择 [Riedmiller et al, 2008],或者每几次与系统的交互以后重新计算策略。例如,在最短路径问题(shortest-path problem)中,每次与系统的交互以后,重新计算路径选择的策略 [Kalyanakrishnan and Stone, 2007; Lange and Riedmiller, 2010a]。在实践中,增长批处理方法是建模时选择将批处理强化学习算法应用到实际系统。从交互的角度看,增长批处理方法与单纯的在线方法非常相似:学习器通过与系统的交互改进策略。从交换的角度来说,在线方法与离线方法的区别并不明显,在线方法也不是区别批处理强化学习的有效指标。谈到批处理强化学习时,更重要的是研究算法和搜索特定更新规则的特点。

增长批处理学习任务

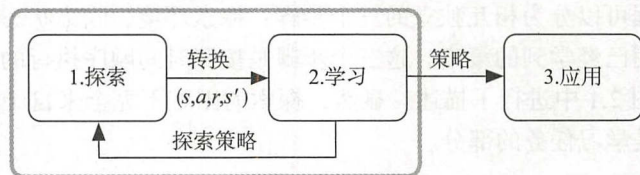


图 2.2 增长批处理强化学习过程包含的三个阶段与图 2.1 所示的“纯”批处理学习过程相同。与单纯的批处理过程不同的是,增长批处理强化学习过程在探索与学习两个阶段之间交替多次,并通过使用中间策略的方式增加存储的转换数目

2.3 批处理强化学习算法的基础

从概念的角度来说,无模型(model-free)的在线学习算法(例如 Q 学习)在处理具有小而离散的状态空间的问题中得到了成功的运用。但是,更现实的系统具有更大的或者连续的状态空间,无模型的在线学习算法运用到这些系统的时候,遇到很多限制因素。可以归纳为以下三个问题:

- 1) 探索开销造成了慢速学习。
- 2) 函数的逼近造成的稳定性问题。
- 3) 随机逼近 (approximation) 造成的低效。

现代批处理强化学习算法的一个共同因素是：这些算法通常解决上述三个问题，并拿出每一个问题的具体的解决方案。接下来，我们将更详细地讨论这些问题，并按照历史顺序提出建议的解决方案，且形成现代批处理强化学习的主要想法。

1. 解决探索开销的“经验重放”想法

为了解释“探索开销”的问题，让我们考虑通常的面向无模型在线学习的 Q-更新规则。Q-更新规则包含两个参数：学习率 α 和阻尼系数 γ ，可表述为

$$Q'(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a' \in A} Q(s',a') - Q(s,a) \right] \quad (2.1)$$

见第1章。在单纯的在线 Q 学习中，每一个时间步骤，学习器在学习与探索之间交替：在状态 s ，学习器选择并执行一个动作 a 。当观察到接下来的状态 s' 和奖励 r ，学习器立即更新价值函数和相应的贪婪策略，具体细节见第1章。根据公式 (2.1)，当学习器完成价值函数的更新后，描述当前状态的数组 (s,a,r,s') 将被清零。然后，学习器将继续探索和更新策略 $Q'(s,a)$ 。虽然这种方法能够保证在极限附近收敛，但是这种方法有严重的性能问题：因为本地更新来自于经验性的转换。例如，在第 t 步的时候，如果学习器更新 Q 值中的状态 s_t 和动作 a_t ，这将影响动作空间 A 中在上一个时间点的动作 a 的状态值 s_{t-1} 。但是，因为学习器仅仅在下次访问时更新前面的状态 s_t ，这个更改不会立即反馈给所有涉及的前面的状态。状态 s_{t-1} 前面的所有状态需要另一个操作来更新状态的数值。这个性能问题是，这些交互不必从系统收集更多的有价值的信息，但是，在时间链上的所有状态需要按照从晚到早的顺序依次更新。实际上，Q 学习中的许多交换采用的是这种“传播”类型的方式。由于无模型的 Q 学习的这些更新是“物理”的，“物理”的含义是学习器需要与系统真实的交互，这些更新也很难根据均匀分布 (uniform distribution) 或者其他分布在整个状态空间中取样。

49

为了克服这个性能上的问题，Lin 提出了“经验重放” (experience replay) 的概念 [Lin, 1992]。虽然“经验重放”的目的是解决在线学习中的“探索开销”问题，回想起来，我们可能会发现“经验重放”是基本的但非第一个用于解决增长批处理问题的技术。经验重放的主要想法是通过以下两种方式加速收敛：一种是通过使用收集到的状态转换；另一种是学习器反复使用经验重放，以便在与系统交互的过程中收集更多新的样本。可以存储一个或者多个被取样的转换样本。每一次与系统交互以后，根据公式 (2.1)，利用已经存储的被取样的转换样本更新每一个已经存储的转换的 Q 函数。虽然基本的在线 Q 学习缺乏已经存储的转换包含的有价值的信息，但是 Q 学习能够通过先前的已经更新的状态反向传播 (back-propagate) 这些有价值的信息。通过经验重放收集的转换类似于各个状态之间的连接，它会通过沿着连接传播有价值的信息来更有效地利用这些信息，并且在理想情况下加快收敛的速度。

2. 解决稳定性问题的“拟合”想法

在线的强化学习经常使用异步更新这一策略，而异步更新 (asynchronous update) 的含义是：在每次取样以后，价值函数 (Value Function, VF) 立即依照当前的状态进行本地更新 (locally update)，而不改变所有其他状态的数值。在离散的情况下，异步更新的含义是，在 Q 学习过程中，每个状态-动作对 (s,a) 的 Q 值将被立即更新，也就是说，转换的起始状

- 50 态的值被立即覆盖 (over-writing) 了。随后的更新将使用这些更新值进行相应的更新。这个想法也可用于函数的逼近, 首先执动作态规划更新:

$$\bar{q}_{s,a} = r + \gamma \max_{a' \in A} f(s', a') \quad (2.2)$$

通过添加当前的奖励和随后状态的拟合值, 重新计算当前的状态-动作对 (s, a) 的逼近值 $\bar{q}_{s,a}$ 。然后通过按照新计算的状态-动作对 (s, a) 的逼近值 $\bar{q}_{s,a}$ 调整拟合值, 并立即存储逼近方程的新值。

$$f'(s, a) \leftarrow (1 - \alpha)f(s, a) + \alpha \bar{q}_{s,a} \quad (2.3)$$

请注意, 通过任意一个函数的逼近算法, 计算真实的 Q 值方程 Q^* 的拟合值 f' , 重新排列公式 (2.1), 得到公式 (2.2) 和公式 (2.3)。

[Baird 1995, Gordon 1996] 发现, Q 学习与函数逼近的某种特定形式的组合将导致不稳定甚至发散 (divergence)。只有逼近算法和更新规则的特定组合, 或者在系统和奖励结构的特定情况和假设之下, 才能产生稳定的数值动作 [Schoknecht and Merke, 2003]。让特定的学习算法在系统中工作, 这需要丰富的工程经验。稳定性问题通常来自于两个方面: 函数的逼近过程中产生的相互关联的错误, 以及最优价值函数与预估的价值函数之间的偏差。当动态规划 (Dynamic Programming, DP) 逐步地缩小 $Q(s, a)$ 和最优 Q 函数 $Q^*(s, a)$ 之间的差距, 将更新的值存储在函数的逼近中 (公式 (2.3)), 这可能带来更大的误差。此外, 这种逼近误差将影响所有后续的动态规划更新, 还可能造成不收敛的情况。当使用全局函数逼近的时候, 例如多层感知机 (multi-layer perceptron), 这个问题会变得更加糟糕 [Rumelhart et al, 1986; Werbos, 1974]。提高状态-动作对 (s, a) 的单个 Q 值可能损害整个状态空间中所有其他的拟合值。

在这种情况下, Gordon 提出来非常吸引人的想法: 轻微地改变更新算法, 使动态规划步骤与函数的逼近步骤分开。Gordon 的主要想法是: 首先, 将动态规划更新运用到一个叫“支持”集合的所有成员中 (其中, 支持集合的含义是分布在状态空间中的多个点的集合), 运用公式 (2.2) 计算所有支持集合的目标值; 然后, 将这些已经更新过的目标值作为训练集, 训练 (拟合) 一个函数的逼近模型, 并根据公式 (2.3) 取代本地的更新值 [Gordon, 1995a]。在这个意义上说, 估计出来的 Q 函数的更新是同步的, 更新在所有支持集合中同时发生。虽然 Gordon 介绍的是基于模型的值迭代的拟合方法, 但这种方法是所有现代批处理算法的基础和出发点。

- 51 3. 取代低效的随机逼近

Gordon 讨论了将“拟合”想法迁移到无模型的基于取样的方法 (例如 Q 学习) 的可能性。但是, 他没有发现解决的方案, 也没有从附近样本中支持集的元素值中辨别出收敛的问题。Ormoneit 和 Sen 最后终于发现了如何将他们的想法运用到基于样本的模型。Ormoneit 和 Sen 建议: 不使用状态空间中任意选择的支持集合来拟合价值函数, 而是通过基于核的拟合方法和在转换的起始状态或者终止状态使用已经采样的转换来拟合价值函数 [Ormoneit, Sen 2002]。Ormoneit 和 Sen 的想法是: 在实际采样的转换中, 计算每一个状态-动作对的估计值, 即奖励加上下一个状态的期望值; 然后, 通过求附近转换的平均值的方法计算每一个状态-动作对的估计值。从技术上讲, 非精确的随机操作能够取代 Gordon 开发的精确的动态规划操作, 正式的定义在 2.4.1 节。这个随机算子 (operator) 没有使用动态规划步骤中的精确模型。相反, 通过从未知的转换模型中取得的随机转换样本, 这个算子可以估计出

态规划操作的准确值。[Lagoudakis and Parr, 2001, 2003] 也独立提出了一个类似的想法。

伴随而来的另一个作用是，通过存储的样本平均值来计算特定转换的开销和估计状态和动作的值，并解决与常规的 Q 学习中的随机逼近问题相关的另一个性能问题。在基于模型的值迭代中，根据以后状态的可能值，当前的状态值能够在一个步骤内更新。在 Q 学习中，因为随机逼近 (stochastic approximation) 替代了模型，这样的一次更新需要多次访问状态。此外，由于我们对整个状态空间使用同一个学习率，按照访问的次数，这个学习率不可能对所有的状态都是最优的。实际上，这使得达到最佳收敛率 (convergence rate) 是不可能的。Ormoneit 和 Sen 的算法不依赖于随机逼近 (stochastic approximation)，而是通过计算取样的转换的平均值，隐式地估计转换模型 (transition model)。如果转换是在与系统交换的过程中采集的，采集的样本形成了一个真实分布的随机采样。

2.4 批处理强化学习算法

Ormoneit 和 Sen 提出了统一的基于核的算法，而这个算法是本书后面章节提出的多种算法的统一基础学习框架。基于核的近似动态规划是经验重放、拟合和基于核的自逼近 (kernel-based self-approximation) 的组合。其中，经验重放的含义是经验的储存和重用，拟合的含义是将动态规划算子与逼近区别开来，而基于核的自逼近是基于采样的。

52

2.4.1 基于核的近似动态规划

Ormoneit 的基于核的近似动态规划 (KADP) 求解一个精确的贝尔曼方程的近似版本：

$$V = HV$$

也可以表达成

$$\hat{V} = \hat{H}\hat{V}$$

以上方程的左边是真实的状态值方程 V 的近似值 \hat{V} ，这与 Gordon 的拟合值迭代非常相似，方程的右边是动态规划算子的准确值 H 的近似值 \hat{H} 。

求解以上方程的 KADP 算法的工作原理如下：从状态值方程的任意起始近似值 \hat{V}^0 开始，基于核的近似动态规划算法的第 i 个循环需要求解如下方程

$$\hat{V}^{i+1} = H_{\max} \hat{H}_{\text{dp}}^a \hat{V}^i$$

对于 p 的转换 (s, a, r, s') 的一个给定的集合

$$\mathcal{F} = \{(s_t, a_t, r_{t+1}, s_{t+1}) | t = 1, \dots, p\}$$

在这个等式中，近似的动态规划算子

$$\hat{H} = H_{\max} \hat{H}_{\text{dp}}^a$$

分成两个部分：准确值 H_{\max} 和近似随机算子 \hat{H}_{dp}^a 。其中，准确值 H_{\max} 对所有动作最大化，近似随机算子 \hat{H}_{dp}^a 对所取样的转换中的每一个动作的动态规划步骤的真实值进行近似计算。根据基于样本的动态规划更新计算第一部分

$$\hat{Q}_a^{i+1}(\sigma) := \hat{H}_{\text{dp}}^a \hat{V}^i(\sigma) = \sum_{(s, a, r, s') \in \mathcal{F}_a} k(s, \sigma) [r + \gamma \hat{V}^i(s')] \quad (2.4)$$

使用加权核 $k(\cdot, \delta)$ ，加权平均的 Q 更新值可以如下计算：

$$r + \gamma \hat{V}^i(s') = r + \gamma \max_{a' \in A} \hat{Q}^i(s', a')$$

与公式 (2.4) 相同, 所有的转换 $(s, a, r, s') \in \mathcal{F}_a$, 其中 $\mathcal{F}_a \in \mathcal{F}$ 是集合 \mathcal{F} 的一个子集。集合 \mathcal{F} 仅仅包含转换 $(s, a, r, s') \in \mathcal{F}$, 这个集合仅使用特定的动作 a 。通过如下方式选择加权核: 距离越远的样本的权重越小, 距离越近的样本的权重越大。

53 等式的第二部分将最大化算子 H_{\max} 运用到计算 Q 函数的估计值 \hat{Q}_a^{i+1} :

$$\hat{V}^{i+1}(s) = H_{\max} \hat{Q}_a^{i+1}(s) = \max_{a \in A} \hat{Q}_a^{i+1}(s) \quad (2.5)$$

请注意, 对于每一个动作 $a \in A$, 为了拟合 Q 函数 $\hat{Q}_a^{i+1}: S \times A \rightarrow R$, 这个算法使用了单一逼近 $\hat{Q}_a^{i+1}: S \rightarrow R$ 。此外, 跟直觉有点相反的是, 在实际的应用中, 公式 (2.5) 的最后一个等式实际上是估计和存储所有转换 $(s, a, r, s') \in \mathcal{F}$ 的所有结束状态 s' , 而不是起始状态 s 。存储结束状态的原因可以这样解释: 我们仅仅使用价值函数在转换的结束状态的值, 而不使用开始状态的有价值的信息, 参见 [Ormoneit, Sen 2002]。更清晰的描述见图 2.3。

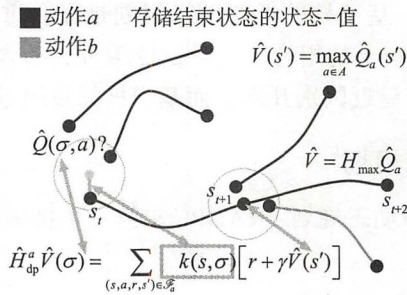


图 2.3 KADP 中基于核的逼近的示意图。计算任意状态 $\delta \in S$ 的 Q 值 $\hat{Q}(\delta, a) = \hat{H}_{dp}^a(\delta) \hat{V}(\delta)$, 基于核的近似动态规划使用附近转换 (s, a, r, s') 的开始状态 s 仅为了计算权重因子 $k(s, \delta)$, 而实际上要根据结束状态 s' 的状态值 $\hat{V}(s')$ 求得 Q 值, 在描述的样本中, $s = s_t, s' = s_{t+1}$)

通过公式 (2.4) 和公式 (2.5) 的迭代, 并通过计算一系列的近似值 $\hat{V}^{i+1}, i = 0, 1, 2, \dots$, 当在转换 (s, a, r, s') 的终点 s' 显式地存储 \hat{V}^i 的 $\hat{V}^i(s')$ 的时候, 当使用满足“平均值”限制的加权核的时候, 基于核的近似动态规划算法最终收敛到唯一解 \hat{V}^i [Gordan, 1995a]。为了满足“平均值”的限制, 权重需要满足以下两个条件。

1) 累加和和:

$$\sum_{(s, a, r, s') \in \mathcal{F}_a} k(s, \sigma) = 1 \quad \forall \sigma \in S$$

2) 非负 (non-negative):

$$k(s, \sigma) \geq 0 \quad \forall \sigma \in S \quad \forall (s, a, r, s') \in \mathcal{F}_a$$

存储和遍历所有状态值而不是所有状态-动作对的决定需要再次运用动态规划步骤, 从基于核的近似动态规划算法推导出一个贪婪策略:

$$\begin{aligned} \pi^{i+1}(\sigma) &= \arg \max_{a \in A} \hat{H}_{dp}^a V^i(\sigma) \\ &= \arg \max_{a \in A} \sum_{(s, a, r, s') \in \mathcal{F}_a} k(s, \sigma) [r + \gamma \hat{V}^i(s')] \end{aligned}$$

从效率的角度看, 这只是一个小程序。虽然 Q 学习需要大量复杂的计算, 而且 Q 学习

在运用的过程中需要存储所有的转换，但这些都还不是根本上的大问题。将动态规划算子运用到固定点 $\hat{V} = \hat{H}\hat{V}$ ，除了带来同一个固定点外，不会带来任何其他变化。

2.4.2 拟合 Q 迭代

批处理强化学习的最流行的实现算法是 Damien Ernst 的拟合 Q 迭代 [Ernst et al, 2005a]。这个算法可视为 Q 学习的批处理强化学习，因为它确实是从基本的 Q 学习更新规则转移到批处理的情况。批处理强化学习开始运行之前，需要满足以下条件：

- 1) 给定 p 个转换 (s, a, r, s') 的固定集 $\mathcal{F} = \{(s_t, a_t, r_{t+1}, s_{t+1}) | t=1, \dots, p\}$ 。
- 2) 一个初始的 Q 值 \bar{q}^0 ，例如 [Ernst et al, 2005a] 中使用 $\bar{q}^0 = 0$ 。
- 3) 对于所有 $(s, a) \in S \times A$ ，设置 Q 函数的起始估计值 $\hat{Q}^0(s, a) = \bar{q}^0$ 。

批处理强化学习执行以下两步循环：

- 1) 从模式 $(s, a; \bar{q}_{s,a}^{i+1}) = \bar{q}^0$ 的一个空白的集合 P^{i+1} 开始。对于每一个转换 $(s, a, r, s') \in \mathcal{F}$ ，根据公式 (2.6)，计算一个新的 Q 值 $\bar{q}_{s,a}^{i+1}$ ：

$$\bar{q}_{s,a}^{i+1} = r + \gamma \max_{a' \in A} \hat{Q}^i(s', a') \quad (2.6)$$

与公式 (2.2) 类似，将一个对应的模式 $(s, a; \bar{q}_{s,a}^{i+1})$ 加入模式集合，得到：

$$P^{i+1} \leftarrow P^{i+1} \cup \{(s, a; \bar{q}_{s,a}^{i+1})\}$$

- 2) 利用监督学习，在模式集合 P^{i+1} 上训练函数拟合。 $i+1$ 步动态规划以后，获得的函数值 \hat{Q}^{i+1} 是 Q 函数的一个近似值。

最初，Ernst 提出了用于价值函数逼近的随机树算法。当它们的结构固定以后，这些决策树算法可以通过基于核的平均数的形式表示出来。因此，第二步可以简化表示为：

$$\hat{Q}_a^{i+1}(\sigma) = \sum_{(s, a; \bar{q}_{s,a}^{i+1}) \in P_a^{i+1}} k(s, \sigma) \bar{q}_{s,a}^{i+1} \quad (2.7)$$

55

其中决策树的结构决定权重 $k(\cdot, \sigma)$ 。对于每一个单独的动作 $a \in A$ ，拟合 Q 迭代的这种衍生算法计算出一个单独的拟合值 \hat{Q}_a^{i+1} ，并满足 $\hat{Q}^{i+1}(s, a) = \hat{Q}_a^{i+1}(s)$ 的条件 [Ernst et al, 2005a]。除了拟合 Q 迭代的这一个衍生算法，Ernst 提出了一个具有连续动作的衍生算法，详见 [Ernst et al, 2005a]。

从理论的角度来说，拟合 Q 迭代仍然是基于 Ormonet 和 Sen 的理论学习框架。根据公式 (2.6) 和公式 (2.7)，拟合 Q 迭代和基于核的近似动态规划具有明显的相似性：

$$\hat{Q}^{i+1}(\sigma, a) = \hat{Q}_a^{i+1}(\sigma) = \sum_{(s, a; \bar{q}) \in P_a^{i+1}} k(s, \sigma) \bar{q}_{s,a}^{i+1} \quad (2.8)$$

$$= \sum_{(s, a, r, s') \in \mathcal{F}_a} k(s, \sigma) \left[r + \gamma \max_{a' \in A} \hat{Q}_a^i(s') \right] \quad (2.9)$$

在拟合 Q 迭代的离散动作中，公式 (2.8) 是公式 (2.7) 的平均步骤。当插入拟合 Q 迭代的动态规划步骤以后，公式 (2.6) 变成公式 (2.9)。通过将公式 (2.5) 插入公式 (2.4) 可以看出公式 (2.9) 的结果与基于核的近似动态规划更新几乎相同

$$\begin{aligned}\hat{Q}_a^{i+1}(\sigma) &= \sum_{(s,a,r,s') \in \mathcal{D}_a} k(s,\sigma) [r + \gamma \hat{V}^i(s')] \\ &= \sum_{(s,a,r,s') \in \mathcal{D}_a} k(s,\sigma) \left[r + \gamma \max_{a' \in A} \hat{Q}_{a'}^i(s') \right]\end{aligned}$$

除了连续处理动作以外，拟合 Q 迭代与基于核的近似动态规划的另一个区别是：算子的分离和显性表示的值。在动态规划步骤（公式（2.4））中，基于核的近似动态规划显性地表示和利用状态的值。在动态规划步骤（公式（2.6））中，拟合 Q 迭代显性地表示 Q 函数，并通过最大化动作的方式计算状态值 $\hat{V}^i(s) = \max_{a \in A} \hat{Q}^i(s, a)$ 。像标准的基于核的近似动态规划提出的那样，尽管在拟合 Q 迭代中允许基于核平均的“懒惰学习”，为了显性地存储 Q 函数，Ernst 假设使用已训练的均衡器和其他参数方程拟合器。而参数方程拟合器的结构无需与起始点（starting point）关联，也无需与转换的终点相关联。因为拟合 Q 迭代显性地代表 Q 函数，拟合 Q 迭代中贪婪策略（greedy policy）的推导是简单的。其中， \hat{Q}^i 是显性实现的，即可以通过简单的（连续动作）函数逼近：

$$\pi^i(s) = \arg \max_{a \in A} \hat{Q}^i(s, a)$$

或者通过对动作 $a \in A$ 的一系列函数逼近 \hat{Q}_a^i 。

因为拟合 Q 迭代与其他类似的算法有轻微的差别，拟合 Q 迭代与在线的 Q 学习更加直接和更加相似。这一点也许能够解释拟合 Q 迭代拥有更大的适应性的原因。

56

2.4.3 基于最小二乘的策略迭代

基于最小二乘的策略迭代 (LSPI) [Lagoudakis and Parr, 2003] 是另外一个批处理模式强化学习算法的例子。与其他在本节综述过的算法相比，基于最小二乘的策略迭代将控制问题的解决融入了策略迭代的学习框架 [Sutton and Barto, 1998]，算法在策略评估与策略改进的步骤之间循环往复。然而，基于最小二乘策略的迭代从不显式地存储策略。相反，基于最小二乘的策略迭代在状态-动作价值方程 Q 的基础上工作。通过状态-动作值方程 Q ，贪婪策略可以利用 $\pi(s) = \arg \max_{a \in A} Q(s, a)$ 公式求解出来。为了表示状态-动作价值函数，LSPI 利用 k 个预定义的基函数（basis function） $\phi_i: S \times A$ 和一个权重向量 $w = (w_1, \dots, w_k)^T$ ，定义了一个参数线性逼近的架构。因此，任何一个 LSPI 学习框架下的逼近的状态-动作价值方程 \hat{Q} 的公式如下：

$$\hat{Q}(s, a; w) = \sum_{j=1}^k \phi_j(s, a) w_j = \Phi w^T$$

相应的策略评估步骤采用面向状态-动作价值函数的最小二乘时序差分学习算法，称为 LSQ [Lagoudakis and Parr, 2001] 或 LSTDQ [Lagoudakis and Parr, 2003]。这种算法将转换 (s, a, r, s') 的有限集合 \mathcal{D} 输入当前的策略方程 π_m 中，正如前面所提到的，这种算法采用一系列权重所代表的当前策略 π_m ，这些算法决定一个价值函数 \hat{Q} 。通过这些输入，在转换集合决定的状态分布下，最小二乘时序差分学习算法为给定的策略推导出状态-动作价值方程。毫无疑问，最小二乘时序差分学习算法返回了价值函数 \hat{Q}^{π_m} ，在以上描述的线性结构下，价值函数通过权重向量 w^{π_m} 完全描述。

总体来说，寻找到的价值函数是 \hat{H}_π 算子的固定点

$$(\hat{H}_\pi Q)(s, a) = r(s, \pi(s)) + \gamma \sum_{s' \in S} p(s, \pi(s), s') \max_{b \in A} Q(s, b) \quad (2.10)$$

例如， $\hat{H}_{\pi_m} Q^\pi = Q^{\pi_m}$ 。因此，一个好的拟合值 \hat{Q}^{π_m} 应该服从方程 $\hat{H}_{\pi_m} \hat{Q}^{\pi_m} \approx Q^{\pi_m} = \Phi (w^{\pi_m})^\top$ 。实际上，最小二乘时序差分学习算法的目标是发现一个向量 w^π ，而这个向量 w^π 是将 \hat{H}_{π_m} 算子运用于 \hat{Q}_{π_m} ，使其与真实的 L_2 范数极小化方式的结果越来越近。为此，最小二乘时序差分学习算法使用了一个正交投影 (projection) 和集合

$$\hat{Q}^{\pi_m} = \left(\Phi (\Phi^\top \Phi)^{-1} \Phi^\top \right) \hat{H} Q^{\pi_m} \quad (2.11)$$

通过公式 (2.10) 的精简形式， $\hat{H}_{\pi_m} Q^{\pi_m} = \mathcal{R} + \gamma \mathcal{P} \Pi_{\pi_m} Q^{\pi_m}$ ，公式 (2.11) 可以表述成如下形式：

$$w^{\pi_m} = \left(\Phi^\top (\Phi - \gamma \mathcal{P} \Pi_{\pi_m} \Phi) \right)^{-1} \Phi^\top \mathcal{R}$$

57

其中， Π_{π_m} 是一个规模为 $|S| \times |S||A|$ 的随机矩阵，这个随机矩阵描述策略 π_m ：

$$\Pi_{\pi_m}(s, (s', a')) = \pi_m(s', a')$$

重要的是在给定样本集合 \mathcal{S} 的基础上，最小二乘时序差分学习算法等式拟合系统的模型，即 P 是一个规模为 $|S||A| \times |S|$ 的包含转换概率的随机矩阵，而转换概率是根据公式 P 从转换集合中采样得到的：

$$P((s, a), s') = \sum_{(s, a', s') \in \mathcal{S}} 1 / \sum_{(s, a', \cdot) \in \mathcal{S}} 1 \approx \Pr(s, a, s')$$

其中， \mathcal{R} 是一个规模为 $|S||A|$ 的向量，而向量 \mathcal{R} 总结了集合 \mathcal{S} 中的奖励。当为当前策略 π_m 计算出状态-动作函数 \hat{Q}^{π_m} 以后，通过以下公式能够推导出贪婪 (改进) 策略 π_{m+1} 。

$$\pi_{m+1}(s) = \arg \max_{a \in A} \hat{Q}_m^\pi(s, a) = \arg \max_{a \in A} \phi(s, a) (w^{\pi_m})^\top$$

因为 LSPI 并不显性地存储策略，而是隐性地利用基函数和相应的权重 (实际上，利用状态-动作价值函数)，LSPI 的策略改进步骤只通过最小二乘时序差分学习算法用现在的权重向量覆盖旧的权重。

通过比较拟合 Q 迭代算法的一个循环和基于最小二乘的策略迭代算法的一个循环 (一个策略评估和改进步骤)，可以获得有价值的信息。基于最小二乘的策略迭代与以价值函数为中心的拟合 Q 迭代算法的主要区别是，在单个循环中，基于最小二乘的策略迭代为当前的策略计算一个状态-动作价值函数 Q^{π_m} 的拟合值。因为线性函数拟合架构的特性，不进行 \hat{H}_{π_m} 算子循环，基于最小二乘的策略迭代也能够提供解析解。相反，为了有监督地拟合一个函数拟合器，拟合 Q 迭代算法依赖于一系列的目标值，而这一系列的目标值是基于一个简单的动态规划的更新步骤，即一个简单的 \hat{H} 算子。因此，如果我们从策略迭代的角解解释拟合 Q 迭代算法，这类算法实现了优化策略循环的一类变体，而基于最小二乘的策略迭代实现了标准的 (非优化的) 策略循环。

2.4.4 识别批处理算法

这里描述的算法可以视为现代批处理强化学习的基础，以前的一些其他算法可以视为批

处理算法或者半批处理算法 (semi-batch algorithm)。而且, 在线、离线、半批处理和批处理之间没有绝对的分界线。对于这个问题来说, 最少有两个不同的观点。图 2.4 提出了在线、半批处理、增长批处理和批处理强化学习算法的次序。

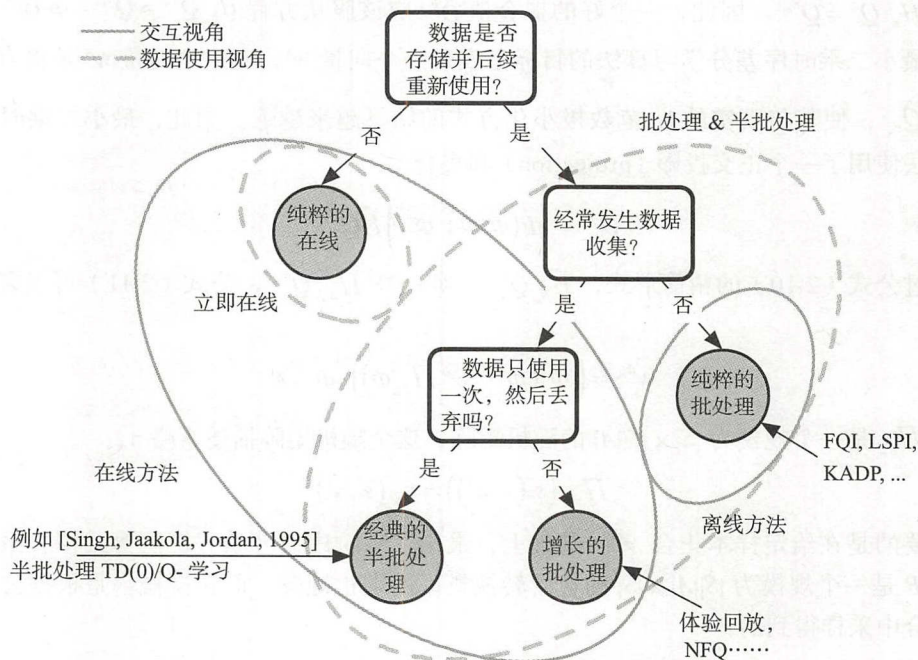


图 2.4 批处理算法与非批处理算法的分类。从交互视角和数据使用的角度来看, 至少有两种不同的视角来定义范畴的边界

在以上算法次序的一边, 我们有纯在线算法, 比如经典的 Q 学习。而在另一边, 我们有纯的批处理算法, 这种批处理算法在一个固定转换集合的基础之上完全“离线”工作。在以上两种绝对的算法之间有许多其他的算法。根据视角的不同, 这些算法可以分为在线算法、批处理算法或半批处理算法。例如, 增长批处理算法可以作为一个在线的方法, 就像其他在线方法一样与系统交互, 并根据新的经验不停地改进策略; 从数据利用的角度来说, 因为其存储所有的经验并使用批处理方法从观察中学习, 可以视为批处理算法。像基于核的近似动态规划算法和基于最小二乘策略的循环算法一样, 虽然 Ernst 提出的拟合 Q 迭代 (类似 KADP 和 LSPI) 是一个在固定样本集合上工作的纯粹的批处理算法, 像 [Kalyanakrishnan and Stone, 2007] 示例的那样, FQI 能够适应增长批处理集合。对于所有纯的批处理算法来说, 这些都是正确的。从另外一个方面来说, 通过增长批处理设置的神经拟合 Q 迭代算法 (见 2.6.1 节), 可以通过直接的方式符合批处理设置。从 20 世纪 90 年代开始, 半批处理算法发展起来 [Singh et al, 1995]。因为半批处理算法对一些转换进行集合式的更新, 所以它不是纯粹进行立即更新的在线算法。半批处理算法在更新完成以后存储和再利用经验, 所以半批处理算法也不是采用完全的批处理方式。

2.5 批处理强化学习理论

批处理强化学习算法引人注目的特点是, 这些算法可以提供稳定的动作规则和类似 Q

学习的更新规则，还可以提供用于不同系统的各类函数拟合器（approximator）。有两个方面需要仔细考虑：稳定性和质量。

[Gordon, 1995a,b] 提出了“平均”这一重要概念，通过显示这类方法的非昂贵的特性，证明了他的模型的收敛性，并依赖马尔可夫动态规划（MDP）算法的经典的带折扣奖励的收缩参数（contraction argument）[Bertsekas and Tsitsiklis, 1996]。对于非折扣的问题，Gordon 确定了更严格的一类兼容的函数拟合器，并证明了自加权平均的收敛性 [Gordon, 1995b]。Ormoneit 和 Sen 将以上证明扩展到无模型的情况，基于核的拟合器与 Gordon 提到的平均值是等价的。拟合值（approximated value）必须是样本的加权平均值，其中所有的权重都是正的，加起来等于 1（见 2.4.1 节）。这些都要求在最大范数下赋予非膨胀性。对于任意给定的样本集合来说，基于最大模（maximum norm）的逼近函数，Ormoneit 和 Sen 表明了他们的随机动态规划算子将会收敛于可能的逼近函数空间的固定点。对于 MDP 算法来说，这些证明通过数量缩小的回报 [Ormoneit and Sen, 2002] 和平均值问题 [Ormoneit and Glynn, 2001, 2002] 来实现。

另一个重要方面是算法发现的解决方案的质量。Gordon 对拟合值迭代不动点的距离函数的最优值给出了绝对值上界（absolute upper bound）[Gordon, 1995b]。这种结合主要取决于函数拟合器及其与最优价值函数的近似性表现（expressiveness）。与函数拟合器不同的是，在无模型的批处理强化学习中，对转换的随机采样方式显然是影响解决方案质量的另一个重要方面。因此，对于 KADP 算法而言，对特定的函数拟合器，拟合解的距离没有绝对值上界。Ormoneit 和 Sen 证明了他们算法的随机一致性（stochastic consistency）。实际上，这是一个更强的表述（statement）。在一定的假设下，不断增加样本的大小可以保证随机收敛到最优价值函数 [Ormoneit and Sen, 2002]。除了对取样转换的其他限制以外，这些假设还包括奖励函数（需要包含 s 、 a 和 s' 的 Lipschitz 连续函数）的平滑限制和核。在整个实验中，使用的一个特定内核满足这些来自“母核”（mother kernel）的约束 [Ormoneit and Glynn, 2001]。

60

$$k_{\mathcal{F}_{a,b}}(s, \sigma) = \phi^+ \left(\frac{\|s - \sigma\|}{b} \right) / \sum_{(s_i, a_i, r_i, s'_i) \in \mathcal{F}_a} \phi^+ \left(\frac{\|s_i - \sigma\|}{b} \right)$$

其中， ϕ^+ 表示单变量高斯函数。参数 b 控制核的带宽（bandwidth），即影响区域，或者“解析度”。依赖于这样的核，一致性证明的主要思想是：首先，依赖于样本数量的增加，为参数 b 定义一个可以接受的下降率（reduction rate）；在已经定义的下降率的基础上，证明 \hat{v}^k 将会随机收敛到最优价值函数。减少带宽参数 b 可以解释为提高分辨率的逼近拟合器。在极限处，当越来越多的样本可用的时候，隐式估计转换模型与真跃迁概率的期望偏差将会变成零。需要重视的是，提高拟合器的解析度将改进光滑奖励函数的逼近，而不必拟合阶跃函数（step function）。例如，奖励函数的 Lipschitz 约束。

这些结果仅仅在批处理强化学习之内运用了平均值。除了这些结果之外，Antos、Munos 和 Szepesvari 提供更新的理论分析。这些新理论没有覆盖更一般的函数拟合器。然而，这些新理论将在未来为非平均值产生有价值的结果 [Antos et al, 2008]。

2.6 批处理强化学习的实现

在这一节，我们将介绍几种批处理强化学习方法在实际问题中的应用。

2.6.1 神经拟合 Q 迭代

特别是对于多层感知机 (multi-layer perceptron) 而言, 将函数拟合到高精度的能力和将函数从一些训练样本泛化的能力使得神经网络成为一个表示价值函数的吸引人的选择 [Rumelhart et al, 1986 ; Werbos, 1974]。然而, 在经典的在线强化学习设置中, 目前的更新常常对迄今为止所做的努力产生不可预见的影响。相对的, 批处理强化学习极大地改变了这种情况: 通过在所有到目前为止所看到的转换中同步更新价值函数, 毁灭以前努力的效果是可以克服的。这就是神经拟合 Q 迭代 (Neural Fitted Q Iteration, NFQ) [Riedmiller, 2005] 的主要思想和推动力。作为第二个重要后果, 在所有训练实例中的同步更新使得批处理监督学习算法 (batch supervised learning algorithm) 的应用成为可能。特别是在神经拟合 Q 迭代学习框架内, 可以使用自适应监督学习算法 Rprop [Riedmiller and Braun, 1993] 作为拟合的核心步骤。

如图 2.5 所示, 使用神经网络实现批处理强化学习的学习框架是相当直接的。然而, 有一些附加的技巧可以帮助克服通过多层感知机 (multi-layer perceptron) 拟合 Q- 价值函数所产生的问题。

```

NFQ_main() {
  input: a set  $\mathcal{F}$  of transition samples  $(s, a, r, s')$  (same, as used throughout the text)
  output: approximation  $\hat{Q}^N$  of the Q-value function
  i=0
  init_MLP()  $\rightarrow \hat{Q}^0$ ;
  Do {
    generate_pattern_set  $P = \{(input_t; target_t), t = 1, \dots, \#D\}$  where:
       $input_t = s, a,$ 
       $target_t = r + \gamma \max_{a' \in A} \hat{Q}^i(s', a')$ 
    add_artificial_patterns(P)
    normalize_target_values(P)
    scale_pattern_values(P)
    Rprop_training(P)  $\rightarrow \hat{Q}^{i+1}$ 
     $i \leftarrow i + 1$ 
  } WHILE ( $i < N$ )

```

图 2.5 神经拟合 Q 迭代的主要循环

- 缩放输入和目标值是成功的关键, 而且缩放输入和目标值应该总是在使用神经网络时完成。因为所有的训练模式在训练开始时是已知的, 一个明智的缩放比例可以很容易地实现。
- 加入人工训练模式 (在 [Riedmiller, 2005] 中也称为“提示目标”)。由于神经网络是从收集的经验中总结出来的, 如果没有或过少的目标-状态经验与零路径成本包括在模式集中, 则可以观察到网络输出趋向于增大至最大值。克服这个问题的一个简单方法是: 建立一个额外的在目标区域 (goal region) 内的人工模式, 目标值为 0。对于许多问题, 这种方法可以很容易地应用, 是非常有效的。在通常的情况下, 目标区域已知, 运用此方法无额外的知识。
- Qmin-启发式: 正则化 Q 目标值 [Hafner and Riedmiller, 2011]。第二种用于减少增加输出值效果的方法是通过从所有目标值减去最低目标值来执行正则化步骤。这将导致至少有一个训练模式的目标值为 0 的模式集。这种方法的优点在于不需要预先

知道目标区域中的状态的额外知识。

- 光滑即时代价函数 [Hafner and Riedmiller, 2011]。由于多层感知机 (multi-layer perceptron) 基本实现了输入到输出的光滑映射 (smooth mapping)，使用平稳的即时代价函数 (cost function) 是合理的。例如，考虑在目标区域之外的即时代价函数，这个即时代价函数在目标区域之内的代价 (cost) 是 0。这导致了最小的时间控制动作，而这在许多应用中是有利的。然而，相应地，路径成本 (path cost) 有相当清晰的跳跃，这很难用神经网络来表示。用平滑的函数替换这个直接的代价函数，当价值函数拟合值更加平滑的时候，清晰的直接代价函数归纳策略的主要特点是得到了广泛的保护。

2.6.2 控制应用中的神经拟合 Q 迭代算法

将强化学习应用于技术过程的控制尤其吸引人，在出现噪声、非线性或者事先不知道过程动作 (process behavior) 的情况下，强化学习能够自动学习到最优或接近最优的控制策略。因为数据效率，批处理强化学习是这个领域主要的突破。通过与真实系统交互的方式，现在从零开始学习复杂的控制动作是可能的。神经拟合 Q 迭代算法在实际应用中的一些最近的例子是：机器人学习装配一个真正的车极系统平衡 (cart-pole system)、气动装置 (pneumatic device) 的最优位置的及时控制、在半小时的驾驶以后学习准确的驾驶一辆车 [Riedmiller et al, 2007]。

下面简要介绍了中型机器人联赛 (RoboCup MidSize League) 中的机器人的神经运球控制 (neural dribble controller) 的学习过程，详情请见 [Riedmiller et al, 2009]。自动机器人 (见图 2.6) 使用一个全方位驱动的摄像头作为主要传感器，其控制间隔期是 33 毫秒。每一个马达命令包含三个数值： v_y^{target} 、 v_x^{target} 和 v_θ^{target} 。其中， v_y^{target} 表示相对于机器人坐标系的目标前进速度， v_x^{target} 表示目标横向速度 (target lateral speed)， v_θ^{target} 表示目标转动速度 (target rotation speed)。

在转向给定目标时，机器人运球 (dribbling) 意味着能够将球保持在机器人前面。中型机器人联赛竞赛规定：不准简单地抓住球；抓取设备只能覆盖三分之一的球。这是一个具有挑战性的任务：抓取设备必须仔细控制机器人。例如，在机器人转换方向的时候，球不能够离开机器人。

将学习问题建模为具有终端目标状态和终端故障状态的随机最短路径问题 (stochastic shortest path problem)。中间步骤受到恒定成本 0.01^2 的处罚。神经拟合 Q 迭代被用作核心学习算法。批处理训练集合的目标值的计算变成了：

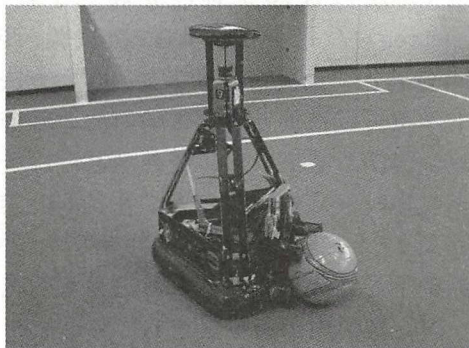


图 2.6 头脑风暴 (Brainstormer) 中型机器人 (实际上，运球的困难在于，根据规则，机器人最多能覆盖到三分之一的球。为了在转弯时不能失去球，因此需要对机器人的运动进行复杂的控制)

$$\bar{q}_{s,a}^{i+1} := \begin{cases} 1.0 & , s' \in S^- \\ 0.01 & , s' \in S^+ \\ 0.01 + \min_{a'} a' \text{in} A \hat{Q}^i(s', a') & , \text{其他} \end{cases} \quad (2.12)$$

其中, S^- 表示机器人失去球的状态。 S^+ 表示机器人拥有球并朝向目标的状态。状态信息 (State information) 包含机器人在相对 x 和 y 方向的速度, 相对于机器人的角速度 x 和 y 。最后, 相对于给定的目标方向的方向。当球的 x 轴大于 50 毫米或者小于 -50 毫米, 或者 y 的相对坐标大于 100 毫米, 机器人遇到失败状态 $s \in S^-$ 。当航向角和目标角的绝对差值小于 5 度时, 就达到了成功状态 (success state)。

机器人由一个表示目标平移和旋转速度的三维动作向量进行控制。使用共有 5 个不同动作的三元组: $U = \{(2.0, 0.0, 2.0), (2.5, 0.0, 1.5), (2.5, 1.5, 1.5), (3.0, 1.0, 1.0), (3.0, -1.0, 1.0)\}$ 。其中, 每一个三元组 (triple) 表示一个 $(v_x^{target}, v_y^{target}, v_\theta^{target})$ 的具体值。

神经拟合 Q 迭代算法是一系列的转换元组 (transition tuple), 格式是 (状态, 动作, 开销, 下一个状态)。其中, 开销的数值来源于外部有价值的信息, 或者通过已知的代价函数 $c: S \times A \times S \rightarrow R$ 即时计算得到的。一个用于采样的常见的程序是在训练 Q 函数和通过贪婪地利用 Q 函数来对转换进行采样之间进行交替。然而, 对于真正的机器人来说, 这意味着每个数据收集阶段之间必须等待新的 Q 函数完成训练。由于把球放回赛场需要人与人之间的交流, 这种情况可能会变得更加严重。因为, 批处理-取样方法可以用来收集多次实验的数据, 而无需再次学习。

价值函数由多层感知机构成。多层感知机包括 6 个状态变量和 3 个动作变量、2 个隐含层 (每层包含 20 个神经元和一个输出神经元 (neuron))。每个批次有 12 次试验, 包含 10 个 NFQ 迭代。运用带标准参数的学习方法, 目标值的学习可以在批处理监督学习的 300 个循环内完成。当学习结束以后, 新控制器可用于在后续数据收集阶段来控制机器人。在我们的实验中, 当完成 11 个批次 132 个实验以后, 一个好的控制器将学习完成。完整的学习过程大约花了一个半小时, 包括用于离线更新的神经近似的 Q 函数的时间。包括准备过程, 与真实机器人的实际的交互时间约为 30 分钟。

如图 2.7 所示, 特别是对于需要转向的空间和时间来说, 神经运球技能 (neural dribbling skill) 明显优于以前使用的人工编码和人工调校的运球程序。从 2007 年开始, 头脑风暴挑战赛 (Brainstormers competition) 开始使用神经运球技巧。通过强化学习, “头脑风暴” 队 2007 年在美国亚特兰大赢得了机器人杯世界挑战赛第一名, 2008 年在中国苏州赢得了机器人杯世界挑战赛第三名。

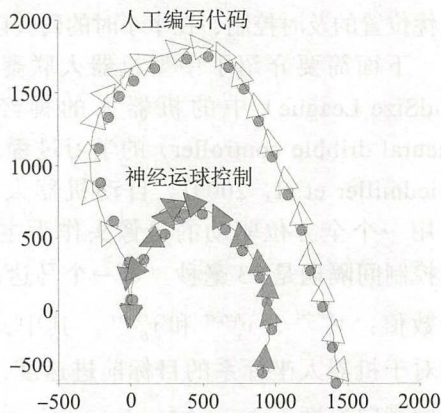


图 2.7 机器人掉头动作的对比: 人工编写代码和神经运球控制。数据来自于真实的机器人。当机器人获得球, 机器人朝着前进的方向 (forward direction) 以每秒 1.5 米至 2 米的速度前进。机器人的位置每 120 毫秒显示一次。机器人通过神经运球控制执行的掉头动作比通过人工编写代码执行的掉头动作要更加快, 转弯半径更加小

2.6.3 面向多学习器的批处理强化学习

前两部分叙述了将批处理模式的数据效率与基于神经网络的函数逼近方案相结合的优

点, 本节阐述批处理方法对协作多学习器强化学习的好处。假设独立学习器, 一个学习器所经历的转换显然受到其他学习器做出的决策的强烈影响。单转换对外部因素的依赖性, 即其他学习器的策略, 引起批处理训练的另一个争论: 虽然单个转换元组包含的有价值的信息可能太少, 无法执行可靠的更新, 但是相当全面的一批经验可能包含足够的有价值的信息, 以便在多学习器的上下文中应用基于价值函数的强化学习。 [65]

去中心化的马尔可夫决策过程 (decentralized Markov decision process) 的学习框架通常用来描述由独立学习器构成的环境, 详见第 15 章或 [Bernstein et al, 2002], 而这些学习器只能读取本地状态的有价值的信息, 不能够读写全局的有价值的信息。这些学习器在动作和学习方面相互独立。找到这些问题的最佳解决方案是: 通过无模型的强化学习为学习器发现近似的联合政策。到此为止, 为每一个学习器 k 定义了一个本地的状态-动作价值函数 $Q_k: S_k \times A_k$, 这个学习器 k 将被连续计算、提高并选择本地动作。

通过直截了当的方式, 一个批处理强化学习算法 (特别是对神经拟合 Q 迭代算法的运用) 能够独立于学习器工作, 忽略可能存在的其他学习器, 并且尝试无需加强它们之间的协调。这个策略可以解释为从相互不协作的学习器中收集带 Q 值的状态-动作对的平均投影 (averaging projection)。因此, 学习器的本地 Q_k 函数低估了最佳合并 Q 函数。以下策略描述了基于批处理强化学习的策略, 这些策略回避这个问题, 并指向了一个更加实际的应用。在这些应用中, 多学习器学习步骤将更加成功地运用, 详见 [Gabel and Riedmiller, 2008b]。

为了更好地估计 Q_k 值, 学习器之间的协调机制 [Lauer and Riedmiller, 2000] 可以在拟合 Q 迭代 (Fitted Q Iteration) 的学习框架内进行应用和集成。最基本的想法是: 每一个学习器最优化的假设是所有其他学习器的动作都是最优的 (尽管由于探索, 它们并不是最优的)。当一个学习器的最优联合动作被执行, 对于价值函数的更新和学习到的策略才能够起作用。这种协调方案 (coordination scheme) 的性能在有噪声的情况下很快就降低了, 这是在收集转换的阶段假设 DEC-MDP 算法状态转换的确定性的原因。然而, 在使用算法所学习到的策略时可以放弃这种假设。

对于多学习器的例子, 修改了 FQI 算法的第一步 (见公式 (2.6)): 每一个学习器 k 收集自己的转换集合 \mathcal{F}_k , 而转换集合 \mathcal{F}_k 拥有本地转换 (s_k, a_k, r_k, s'_k) 。算法创建一个精简的 (也称为“乐观的”) 训练模式集合 \mathcal{O}_k , 满足条件 $|\mathcal{O}_k| \leq |P_k|$ 。给定一个确定的环境, 以及在数据收集阶段将系统重置为特定初始状态的能力。学习器 k 进入学习器 s_k 多于一次的概率大于零。因此, 如果学习器 s_k 多次完成一个特定的动作 $a_k \in A_k$, 因为其他学习器 s_k 选择了不同的本地动作, 学习器 s_k 产生了非常不同的奖励和本地继承状态 k 。除了考虑 \mathcal{F}_k 中的所有元组, 只有那些产生了最大期望回报的学习器才被用来创造 \mathcal{O}_k 。这意味着我们假设所有其他学习器都力所能及地采取了局部的动作。也就是说, 当与 a_k 混合的时候, 学习器基本上适应当前的全局状态。因此, 对于学习器 k 的给定的本地状态-动作对 (s_k, a_k) , 最优化的目标 Q 值 q_{s_k, a_k}^{i+1} 根据如下公式计算: [66]

$$q_{s_k, a_k}^{i+1} := \max_{\substack{(s, a, r, s') \in \mathcal{F}_k \\ s=s_k, a=a_k}} \left(r + \gamma \max_{a' \in A_k} \hat{Q}_k^i(s', a') \right)$$

因此, 对于相同的 s_k 和 a_k 值来说, \mathcal{O}_k 实现了 \mathcal{F}_k 的分割。 q_{s_k, a_k}^{i+1} 是即时奖励的最大和, 对所有数组 $(s_k, a_k, \cdot, \cdot) \in \mathcal{F}_k$, 缩减预期回报。

有许多应用程序存在这类问题, 包括生产设计问题和资源分配问题 [Gabel and

Riedmiller, 2008c]。一类可以归结为去中心化 MDP 算法的特殊问题，是由工作 - 购物时间分配问题构成的 [Brucker and Knust, 2005]。这类问题的目标是将有限的资源（机器）分配一定数量的工作（任务），而任务分配的方式是优化特定的目标。将批处理强化学习方式用于时间分配，一个学习器将分配给一个资源，接收关于需要处理的工作集合的本地有价值的信息，特征值是特征函数逼近的神经网络输入的特征，并决定下一个需要分配的工作。学习器与调度程序不断交互，学习器不断收集相关的批处理转换（batch transition），并通过利用上述神经拟合 Q 迭代适应算法，通过学习器 - 特定状态 - 动作价值函数 \hat{Q}_k ，学习器推导出一个增强的任务分发策略。对于公认的基准问题来说，学习到的分发策略展现出良好的性能，以及良好的扩展能力，这些分发策略能够直接应用于修改后的出厂设置。

2.6.4 深度拟合 Q 迭代

总的来说，现在的强化学习算法仍然局限于解决低维度状态空间中的问题。例如，从高维度的视觉输入中（如相机拍摄的原始图像）直接学习策略是几乎不可能的。在这样的任务中，工程师提供了一个从高维度输入中抽取相关有价值的信息，并将这些有价值的信息编码到低维度的特征空间的方法。然后将该学习算法应用于人工构造的特征空间中。

67

批处理强化学习提供直接处理高维度状态空间的可能性。对于一个系列的转换 $\mathcal{F} = \{(s_t, a_t, r_{t+1}, s_{t+1}) | t = 1, \dots, p\}$ 来说，状态 s 是高维状态空间 $s \in R^n$ 的元素。强化学习的核心思想是使用适当的无监督学习算法，自动从数据中提取特征。理想状态下，在生成的特征向量 $z = \phi(s)$ 中学习到的映射 $\phi: R^n \rightarrow R^m (m \ll n)$ 应该编码状态空间 s 中包含的相关有价值的信息。通过批处理强化学习，依赖于批处理增强方法，我们可以将特征空间的学习与稳定的、高效的算法学习策略结合起来。在增长批处理方法中，当开始一个新的学习阶段，我们首先使用这些数据 \mathcal{F} 学习一个新的特征提取映射 $\phi: R^n \rightarrow R^m$ ，然后在这个特征空间中学习一个策略。首先将所有样本从状态空间映射到特征空间，构建一个在特征空间的模式集 $\mathcal{F}_\phi = \{(\phi(s), a, r, \phi(s')) | (s, a, r, s') \in \mathcal{F}\}$ ，然后施加一个批处理算法，例如 FQI 算法。因为所有的经验都存储在增长批处理方法中，在每次探索之后，都可以改变映射，并利用新的数据改进批处理方法。一个价值函数的新的拟合值能够从映射的转换中计算出来。实际上，通过转换集合 \mathcal{F} ，将 \hat{Q}^ϕ 的特征提取 ϕ 的拟合值“转换”到一个新的特征提取集合 ϕ 而不损失任何有价值的信息。我们简单地将 FQI 算法的一步运用到计算轻微修改的目标值：

$$\bar{q}_{\phi'(s),a} = r + \gamma \max_{a' \in A} \hat{Q}_{a'}^\phi(\phi(s'))$$

当为 $\bar{q}_{\phi'(s),a}$ 计算新的目标值，状态 s 的新特征向量为 $\phi'(s)$ ，这个更新使用来自于下一个状态 s' 的期望奖励，而下一个状态 s' 是旧的拟合值 \hat{Q}^ϕ 通过旧的特征空间中特征向量 $\phi(s')$ 给出的。然后使用这些目标值为新的特征空间计算一个新的拟合值 $\hat{Q}^{\phi'}$ 。

我们已经将这个想法实现为一个新的算法，称为深度拟合的 Q 迭代（DFQ）[Lange and Riedmiller, 2010a,b]。深度拟合 Q 迭代使用一个深度自编码神经网络 [Hinton and Salakhutdinov, 2006]，这个神经网络具有百万级别的权重，这些权重用于从高维度视觉输入无监督地学习低维度的特征空间。这些神经网络训练融入了来自于 FQI 的批处理强化学习的算法，同时允许学习可行的特征空间和有用的控制策略（见图 2.8）。通过依赖于基于核的平均值而逼近在自动创建的状态空间中的价值函数，DFQ 从批处理方法中继承了稳定的学

习动作。通过扩展 Ormoneit 和 Sen 的理论结果，DFQ 的内循环将收敛于来自于任一具有不连续奖励的马尔可夫动态规划（MDP）算法的给定集合的唯一解 [Lange, 2010]。通过合成的图像 [Lange and Riedmiller, 2010b] 或者从屏幕抓取的图像 [Lange and Riedmiller, 2010a]，DFQ 能够成功应用于在网格世界的基准问题中学习控制策略，只通过来自于顶部安装的相机抓取的图像数据 [Lange, 2010] 来控制赛车。

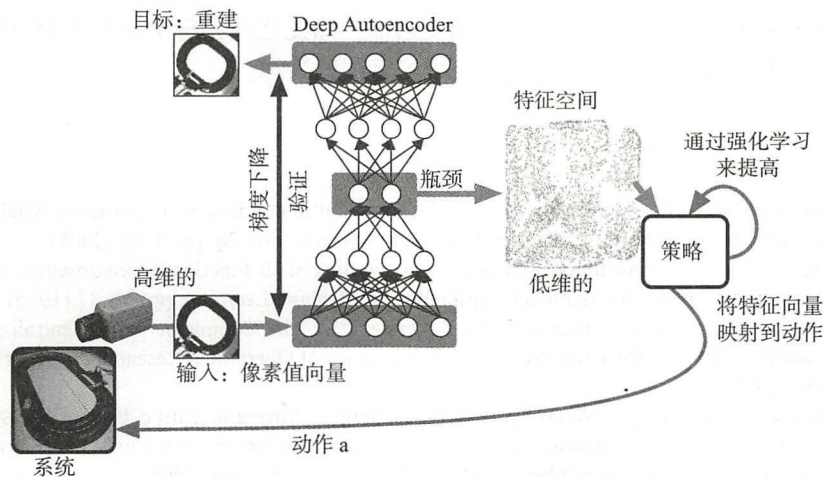


图 2.8 DFQ 模式图示。将来自于系统的原始视觉有价值的信息输入到深层自动编码器神经网络中，学习在其瓶颈层（bottle-neck layer）中提取相关有价值的信息的低维度编码。然后，生成的特征向量用于在批处理更新的情况下学习相关的策略

2.6.5 应用 / 发展趋势

下表简要介绍了批处理强化学习方法的最新应用。

领域	描述	方法	参考文献
技术过程控制，真实世界	槽赛车	NFQ	Kietzmann and Riedmiller (2009)
技术过程控制，真实世界	盘带足球机器人	NFQ	Riedmiller et al (2009)
技术过程控制，真实世界	驾驶自主车	NFQ	Riedmiller et al (2007)
技术过程控制，模拟	非线性控制的基准	NFQ, NFQCA	Hafner and Riedmiller (2011)
技术过程控制，模拟	极点摆动与平衡	Batch RL, Gaussian 过程	Deisenroth et al (2009)
技术过程控制，模拟	移动轮摆	NFQ, FQI (多余的树木)	Bonarini et al (2008)
技术过程控制，模拟	半主动悬挂控制	基于树的批处理	Tognetti et al (2009)
技术过程控制，模拟	电力系统的控制与 MPC 的比较	FQI (多余的树木)	Ernst et al (2005b), Ernst et al (2009)
投资组合管理，模拟	管理金融交易	KADP	Ormoneit and Glynn (2001)
标杆管理，仿真	山地车，使机器人	FQI, CMAC	Timmer and Riedmiller (2007)
多学习器系统仿真	分散的调度策略	NFQ	Gabel and Riedmiller (2008a)
多学习器系统仿真	Keepaway 足球	FQI (NN, CMAC)	Kalyanakrishnan and Stone (2007)
医疗应用	癫痫治疗	基于树的批处理	Guez et al (2008)

2.7 总结

本章回顾了批处理模式强化学习的历史根源和早期算法以及现代算法和应用, 包括基于核的近似动态规划、拟合 Q 迭代、基于最小二乘策略的循环、最小二乘时序差分学习算法、深度拟合 Q 迭代、神经拟合 Q 迭代算法等。近年来, 主要由于批处理方法的核心优点, 即批处理方法有效地利用了收集到的数据, 以及动态规划和价值函数的逼近步骤的分离所得到的学习过程的稳定性这方面的研究活动大幅度增加。除此之外, 现实世界中学习任务的各种实际实现和应用促进了研究者对批处理强化学习算法的兴趣。

参考文献

- Antos, A., Munos, R., Szepesvari, C.: Fitted Q-iteration in continuous action-space MDPs. In: *Advances in Neural Information Processing Systems*, vol. 20, pp. 9–16 (2008)
- Baird, L.: Residual algorithms: Reinforcement learning with function approximation. In: *Proc. of the Twelfth International Conference on Machine Learning*, pp. 30–37 (1995)
- Bernstein, D., Givan, D., Immerman, N., Zilberstein, S.: The Complexity of Decentralized Control of Markov Decision Processes. *Mathematics of Operations Research* 27(4), 819–840 (2002)
- Bertsekas, D., Tsitsiklis, J.: *Neuro-dynamic programming*. Athena Scientific, Belmont (1996)
- Bonarini, A., Caccia, C., Lazaric, A., Restelli, M.: Batch reinforcement learning for controlling a mobile wheeled pendulum robot. In: *IFIP AI*, pp. 151–160 (2008)
- Brucker, P., Knust, S.: *Complex Scheduling*. Springer, Berlin (2005)
- Deisenroth, M.P., Rasmussen, C.E., Peters, J.: Gaussian Process Dynamic Programming. *Neurocomputing* 72(7-9), 1508–1524 (2009)
- Ernst, D., Geurts, P., Wehenkel, L.: Tree-Based Batch Mode Reinforcement Learning. *Journal of Machine Learning Research* 6(1), 503–556 (2005a)
- Ernst, D., Glavic, M., Geurts, P., Wehenkel, L.: Approximate Value Iteration in the Reinforcement Learning Context. Application to Electrical Power System Control. *International Journal of Emerging Electric Power Systems* 3(1) (2005b)
- Ernst, D., Glavic, M., Capitanescu, F., Wehenkel, L.: Reinforcement learning versus model predictive control: a comparison on a power system problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 39(2), 517–529 (2009)
- Gabel, T., Riedmiller, M.: Adaptive Reactive Job-Shop Scheduling with Reinforcement Learning Agents. *International Journal of Information Technology and Intelligent Computing* 24(4) (2008a)
- Gabel, T., Riedmiller, M.: Evaluation of Batch-Mode Reinforcement Learning Methods for Solving DEC-MDPs with Changing Action Sets. In: Girgin, S., Loth, M., Munos, R., Preux, P., Ryabko, D. (eds.) *EWRL 2008. LNCS (LNAI)*, vol. 5323, pp. 82–95. Springer, Heidelberg (2008)
- Gabel, T., Riedmiller, M.: Reinforcement Learning for DEC-MDPs with Changing Action Sets and Partially Ordered Dependencies. In: *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, IFAA-MAS, Estoril, Portugal, pp. 1333–1336 (2008)
- Gordon, G.J.: Stable Function Approximation in Dynamic Programming. In: *Proc. of the Twelfth International Conference on Machine Learning*, pp. 261–268. Morgan Kaufmann, Tahoe City (1995a)
- Gordon, G.J.: Stable function approximation in dynamic programming. Tech. rep., CMU-CS-95-103, CMU School of Computer Science, Pittsburgh, PA (1995b)
- Gordon, G.J.: Chattering in SARSA (λ). Tech. rep. (1996)
- Guez, A., Vincent, R.D., Avoli, M., Pineau, J.: Adaptive treatment of epilepsy via batch-mode reinforcement learning. In: *AAAI*, pp. 1671–1678 (2008)
- Hafner, R., Riedmiller, M.: Reinforcement Learning in Feedback Control — challenges and

- benchmarks from technical process control. *Machine Learning* (accepted for publication, 2011), doi:10.1007/s10994-011-5235-x
- Hinton, G., Salakhutdinov, R.: Reducing the Dimensionality of Data with Neural Networks. *Science* 313(5786), 504–507 (2006)
- Kalyanakrishnan, S., Stone, P.: Batch reinforcement learning in a complex domain. In: *The Sixth International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 650–657. ACM, New York (2007)
- Kietzmann, T., Riedmiller, M.: The Neuro Slot Car Racer: Reinforcement Learning in a Real World Setting. In: *Proceedings of the Int. Conference on Machine Learning Applications (ICMLA 2009)*. Springer, Miami (2009)
- Lagoudakis, M., Parr, R.: Model-Free Least-Squares Policy Iteration. In: *Advances in Neural Information Processing Systems*, vol. 14, pp. 1547–1554 (2001)
- Lagoudakis, M., Parr, R.: Least-Squares Policy Iteration. *Journal of Machine Learning Research* 4, 1107–1149 (2003)
- Lange, S.: *Tiefes Reinforcement Lernen auf Basis visueller Wahrnehmungen*. Dissertation, Universität Osnabrück (2010)
- Lange, S., Riedmiller, M.: Deep auto-encoder neural networks in reinforcement learning. In: *International Joint Conference on Neural Networks (IJCNN 2010)*, Barcelona, Spain (2010a)
- Lange, S., Riedmiller, M.: Deep learning of visual control policies. In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2010)*, Brugge, Belgium (2010b)
- Lauer, M., Riedmiller, M.: An Algorithm for Distributed Reinforcement Learning in Cooperative Multi-Agent Systems. In: *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pp. 535–542. Morgan Kaufmann, Stanford (2000)
- Lin, L.: Self-Improving Reactive Agents Based on Reinforcement Learning, Planning and Teaching. *Machine Learning* 8(3), 293–321 (1992)
- Ormoneit, D., Glynn, P.: Kernel-based reinforcement learning in average-cost problems: An application to optimal portfolio choice. In: *Advances in Neural Information Processing Systems*, vol. 13, pp. 1068–1074 (2001)
- Ormoneit, D., Glynn, P.: Kernel-based reinforcement learning in average-cost problems. *IEEE Transactions on Automatic Control* 47(10), 1624–1636 (2002)
- Ormoneit, D., Sen, Š.: Kernel-based reinforcement learning. *Machine Learning* 49(2), 161–178 (2002)
- Riedmiller, M.: Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) *ECML 2005. LNCS (LNAI)*, vol. 3720, pp. 317–328. Springer, Heidelberg (2005)
- Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: Ruspini, H. (ed.) *Proceedings of the IEEE International Conference on Neural Networks (ICNN)*, San Francisco, pp. 586–591 (1993)
- Riedmiller, M., Montemerlo, M., Dahlkamp, H.: Learning to Drive in 20 Minutes. In: *Proceedings of the FBIT 2007 Conference*. Springer, Jeju (2007)
- Riedmiller, M., Hafner, R., Lange, S., Lauer, M.: Learning to dribble on a real robot by success and failure. In: *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 2207–2208 (2008)
- Riedmiller, M., Gabel, T., Hafner, R., Lange, S.: Reinforcement Learning for Robot Soccer. *Autonomous Robots* 27(1), 55–74 (2009)
- Rumelhart, D., Hinton, G., Williams, R.: Learning representations by back-propagating errors. *Nature* 323(6088), 533–536 (1986)
- Schoknecht, R., Merke, A.: Convergent combinations of reinforcement learning with linear function approximation. In: *Advances in Neural Information Processing Systems*, vol. 15, pp. 1611–1618 (2003)
- Singh, S., Jaakkola, T., Jordan, M.: Reinforcement learning with soft state aggregation. In: *Advances in Neural Information Processing Systems*, vol. 7, pp. 361–368 (1995)

- Sutton, R., Barto, A.: Reinforcement Learning. An Introduction. MIT Press/A Bradford Book, Cambridge, USA (1998)
- Timmer, S., Riedmiller, M.: Fitted Q Iteration with CMACs. In: Proceedings of the IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL 2007), Honolulu, USA (2007)
- Tognetti, S., Savaresi, S., Spelta, C., Restelli, M.: Batch reinforcement learning for semi-active suspension control, pp. 582–587 (2009)
- Werbos, P.: Beyond regression: New tools for prediction and analysis in the behavioral sciences. PhD thesis, Harvard University (1974)

策略迭代的最小二乘法

Lucian Buşoniu, Alessandro Lazaric, Mohammad Ghavamzadeh,
Rémi Munos, Robert Babuška, Bart De Schutter

摘要

近似强化学习通过使用逼近器来表示解决方案，解决了大规模连续状态空间下应用强化学习的重要问题。本章主要介绍了如何使用最小二乘法来解决策略迭代问题，这是近似强化学习的一类重要算法。我们讨论三种解决策略迭代的核心——策略评估组件的技术，称为：最小二乘时序差分、最小二乘策略评估和贝尔曼残差最小化。我们从它们的一般数学原理开始介绍这些技术，并充分而详细地介绍算法例子。我们将注意策略迭代的在线变体，并提供一个数值示例，突出代表性离线和在线方法的动作。对于策略评估组件以及整体产生的近似策略迭代，随着执行的样本数量和执行的迭代增长到无穷大，我们保证可以获得较好的性能。我们还提供适用于有限数量的样本和迭代的有限样本结果。最后，我们概述几个扩展和改进的算法。

75

3.1 简介

策略迭代是解决强化学习问题的核心过程，它通过估计策略的价值函数来评估策略。然后使用这些价值函数来寻找新的、改良的策略。在第1章我们介绍了经典的策略迭代方法，它使用了表格（一种价值函数及策略的精确表示）。然而，现实中大多数难题都包含大量甚至无穷多元素的状态和动作空间，这就排除了使用表格来精准表示策略迭代的方法。这样就必须使用近似策略迭代。特别的，近似策略评估（为所考虑的策略构造近似价值函数）是近似策略迭代算法主要的、最有挑战性的组成部分。然而，如何对策略进行表达也是一个有挑战性的问题，通常通过从近似价值函数按需计算策略动作来避免显式表示。

用于近似策略评估的一些最强大最先进的算法使用线性参数化的价值函数，并且通过利用由价值函数满足的贝尔曼方程的线性度而在参数中获得线性方程系统。然后，为了获得近似价值函数的参数，以一次或迭代的方式使用基于采样的最小二乘法方式求解该系统。

由于数值方法求解这样的系统十分高效，使用最小二乘法进行策略评估是高效的。另外，利用策略迭代方法的快速收敛性，可以获得总体快速的策略迭代算法。更重要的是，最小二乘法是样本高效的，即它们随着考虑样本数目的增加而迅速地接近它们的解。强化学习的这个性质十分关键，因为现实中从这样的系统获得数据是非常昂贵的。

在本章中，我们将介绍策略迭代的最小二乘法：在策略评估步骤中所使用的基于最小二乘法的近似策略迭代方法。本章结构如下。3.2节简要介绍了经典策略迭代，进一步阐述了本章的核心内容。3.3节是本章的核心，全面介绍了最小二乘法的策略评估方法。3.4节描述了一个在线学习的应用，它使用了一种称为简单最小二乘策略迭代的算法。3.5节举例说明

了离线和在线最小二乘法策略迭代的步骤。3.6 节提供了关于最小二乘策略评估和由此产生的近似策略迭代的理论。最后，3.7 节概述了几个对最小二乘算法重要的扩展和改进，并和其他一些与最小二乘有关的强化学习算法进行对比。

76

3.2 预备知识：经典策略迭代算法

本节中，我们回顾经典的策略迭代算法和第 1 章中的一些相关理论结果，根据本章内容调整了它们的展示角度。

首先，我们先回顾一些符号。一个马尔可夫决策过程有已知的状态 $s \in S$ 和动作（行为） $a \in A$ ，由随机动态 $s' \sim T(s, a, \cdot)$ 管理。并且利用回报 $r = R(s, a, s')$ 描述即时性能，其中 T 是转换函数， R 是回报函数。目标在于找到价值函数 $V^\pi(s)$ 或 $Q^\pi(s, a)$ 的最大化的最优策略 $\pi^*: S \rightarrow A$ 。为了清楚起见，在本章中我们把状态价值函数记为“V 函数”，从而实现与传统上用于状态-动作价值函数的“Q 函数”一致。我们使用“价值函数”来统称 V 函数和 Q 函数。

策略迭代通过迭代评估和改进策略的方法来进行。在策略评估这一步，找到当前策略的 V 函数或者 Q 函数。然后，在策略改进这一步，基于已有的 Q 函数或 V 函数计算新的、更好的策略。这个过程在下一个迭代过程中继续执行。当状态-动作空间是有限的，并且使用价值函数和策略的精确表示时，策略改进获得比先前策略严格更好的策略，除非当前策略已经是最优的。因此保证策略迭代算法可以在有限次迭代下找到最优的策略。算法 6 给出了使用 Q 函数的经典离线策略迭代算法。

```

1: input initial policy  $\pi_0$ 
2:  $k \leftarrow 0$ 
3: repeat
4:   find  $Q^{\pi_k}$  {policy evaluation}
5:    $\pi_{k+1}(s) \leftarrow \arg \max_{a \in A} Q^{\pi_k}(s, a) \quad \forall s$  {policy improvement}
6:    $k \leftarrow k + 1$ 
7: until  $\pi_k = \pi_{k-1}$ 
8: output  $\pi^* = \pi_k, Q^* = Q^{\pi_k}$ 

```

算法 6 使用 Q 函数的策略迭代方法

在策略评估这一步，Q 函数是 Q^π ，策略 π 可以用贝尔曼方程求出：

$$Q^\pi = B_Q^\pi(Q^\pi) \quad (3.1)$$

77

其中贝尔曼映射（也称为备份算子） B_Q^π 被 Q 函数定义为：

$$[B_Q^\pi(Q)](s, a) = E_{s' \sim T(s, a, \cdot)} \{R(s, a, s') + \gamma Q(s', \pi(s'))\} \quad (3.2)$$

同样的，V 函数 V^π 的策略 π 满足贝尔曼方程：

$$V^\pi = B_V^\pi(V^\pi) \quad (3.3)$$

其中贝尔曼映射 B_V^π 用如下公式定义：

$$[B_V^\pi(V)](s) = E_{s' \sim T(s, \pi(s), \cdot)} \{R(s, \pi(s), s') + \gamma V(s')\} \quad (3.4)$$

需要注意的是 Q 函数和 V 函数的绝对值都被限制为 $V_{\max} = \frac{\|R\|_\infty}{1-\gamma}$ ，其中 $\|R\|_\infty$ 是最大绝对回

报。许多算法可用于计算 Q^π 或 V^π ，例如，基于直接求解公式 (3.1) 或公式 (3.3) 中的线性系统以获得 Q 值（或者 V 值），把贝尔曼方程（等式）转化为迭代赋值的任务，或者变成基于时序差分的无模型估计（model free estimation），详见第 1 章内容。

一旦 Q^π 或 V^π 可用，就可以执行策略改进。在这种情况下则出现一个使用 Q 函数和 V 函数的重要差异。当使用 Q 函数时，策略改进仅涉及操作空间上的最大化：

$$\pi_{k+1}(s) \leftarrow \arg \max_{a \in A} Q^{\pi_k}(s, a) \quad (3.5)$$

（再看一次算法 6）然而，基于 V 函数的策略改进需要一个以 T 或者 R 的形式的模型来研究用下面每个动作所产生的转换：

$$\pi_{k+1}(s) \leftarrow \arg \max_{a \in A} E_{s' \sim T(s, a)} \{R(s, a, s') + \gamma V(s')\} \quad (3.6)$$

这是实践中使用 Q 函数的一个重要因素，特别对于无模型算法。

经典的策略迭代需要价值函数的准确表示，这通常只能通过为每个状态-动作对（在 Q 函数的情况下）或每个状态（ V 函数）存储不同的值来实现。当一些变量有着非常大或者无穷多数量的可能值时，比如，当它们是连续型数值，则无法精确地表示，同时价值函数一定是估计值。本章重点介绍近似策略迭代，或者更具体地说，一类使用最小二乘法进行策略评估的算法。

虽然在大规模状态空间中表示策略是非常有挑战性的，但幸运的是，明确的策略表示常常可以被避免。相反，改进的策略可以按需进行计算，通过在需要一个动作的每一个状态中使用公式 (3.5) 或公式 (3.6)。这就意味着通过价值函数使用隐式的策略表示。接下来，我们关注这个设置，另外要求执行精确的策略改进。也就是说，被返回的动作总是在公式 (3.5) 或公式 (3.6) 中达到极大值。例如，当动作空间是离散的，并且没有包含太多数量的动作时，可以满足上述要求。在这种情况下，可以通过计算所有离散动作的价值函数，并且使用枚举的方式在这些值中找到最大值，从而执行策略改进[⊖]。

78

在下文中，我们将介绍 Q 函数的结果和算法。大多数这样的结果和算法可直接扩展到 V 函数。

3.3 近似策略评估的最小二乘法

本节中我们将讨论如何用最小二乘法来策略评估的问题，首先，在 3.3.1 节中，我们将讨论这些方法背后的高层原则（high-level principle）。然后，我们将逐步前进到方法的实际实现。特别是在 3.3.2 节中，我们推导出用于线性参数逼近情况下的理想化的、基于模型的算法版本。在 3.3.3 节中我们概述了它们的实际的无模型算法实现。为了避免偏离主线，我们把对参考文献的讨论推迟到 3.3.4 节。

3.3.1 主要原则和分类

在那些具有很大或者连续状态-动作空间的问题中，价值函数虽然不能完全被表达，但必须被逼近。因为贝尔曼方程（公式 (3.1)）的解通常不能由近似值代表，贝尔曼方程必须

⊖ 一般来说，当动作空间大或连续时，公式 (3.5) 或公式 (3.6) 的最大化问题很难被精确解决。在这种情况下，只有近似策略改进方法可以解决这个问题。

近似求解。策略评估的两类最小二乘法可以通过它们贝尔曼方程近似解的方法被区分。如图 3.1 中表示的预计策略评估 (projected policy evaluation) 和贝尔曼残差最小化。

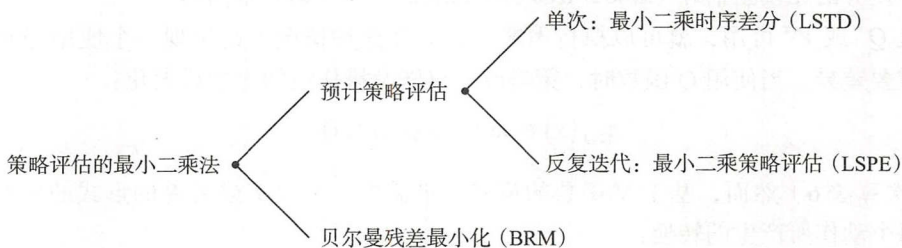


图 3.1 所有算法的分类

预计策略评估的方法寻找一个 \hat{Q} ，近似于备份版本 $B_Q^\pi(\hat{Q})$ 在可被代表的 Q 函数空间上的投影：

$$\hat{Q} \approx \Pi(B_Q^\pi(\hat{Q})) \quad (3.7)$$

其中 $\Pi: \mathcal{Q} \rightarrow \mathcal{Q}$ 表示 \mathcal{Q} 空间中所有 Q 函数到 \mathcal{Q} 空间所有可描述 Q 函数的投影。求解这个方程式等同于最小化 \hat{Q} 和 $\Pi(B_Q^\pi(\hat{Q}))$ 之间的距离^①：

$$\min_{\hat{Q} \in \mathcal{Q}} \|\hat{Q} - \Pi(B_Q^\pi(\hat{Q}))\| \quad (3.8)$$

其中 $\|\cdot\|$ 表示一个通用的范数大小。3.3.2 节将给出一个这样的范数的例子。公式 (3.7) 中的关系叫作投影贝尔曼方程，因此称为预计策略评估。在 3.3.2 节中可以看到，在近似线性公式 (3.7) 情况下，事实上可以获得精确解，例如，具有平等性（而一般情况下，可能不存在一个通过 B_Q^π 和 Π 的函数 \hat{Q} 来完全回到它本身）。

可以进一步确定采用预计策略评估原理的两个子类方法。直接针对投影贝尔曼方程的解决方案的单次迭代方法；使用反复迭代求解的方法。这两种方法叫作最小二乘时序差分 (LSTD) [Bradtke and Barto, 1996; Boyan, 2002] 和最小二乘策略评估 (LSPE) [Bertsekas and Ioffe, 1996]。需要注意的是 LSPE 迭代是内部策略评估的步骤，包含在总体策略迭代算法的大迭代中。

贝尔曼残差最小化 (BRM) 方法不使用投影，但是试图使用下列近似方程来直接求解贝尔曼方程：

$$\hat{Q} \approx B_Q^\pi(\hat{Q})$$

通过执行下列最小化：

$$\min_{\hat{Q} \in \mathcal{Q}} \|\hat{Q} - B_Q^\pi(\hat{Q})\| \quad (3.9)$$

$\hat{Q} - B_Q^\pi(\hat{Q})$ 的差值称为贝尔曼残差，因此得名 BRM。

在下文中，我们将添加后缀“Q”作为 Q 函数的缩略词（例如 LSTD-Q、LSPE-Q、BRM-Q）且在 V 函数的情况下使用后缀“V”。我们将使用首字母缩写来指代这两种类型的

① 这个公式和其他公式一样，也可以使用一个多步版本的贝尔曼映射 (B_Q^π) （或 (B_r^π) ），被标量 $\lambda \in [0, 1)$ 参数化，在本章中，我们主要考虑单步执行的情况，即 $\lambda = 0$ ，但是我们简要地在 3.7 节中讨论多步的情况。

方法。总的来说，它们之间的区别并不重要。

80

3.3.2 线性情况下和矩阵形式的方程

直到现在，价值函数的逼近器，以及公式(3.8)和公式(3.9)的最小化的规范仍然没有被指定。最常见的选择分别是近似的线性参数和平方、加权的欧几里得范数。通过这些选择，由投影和 BRM 策略评估解决的最小化问题可以用矩阵和向量来表示，并且具有封闭式的解决方案，最终可以得到有效的算法。这类方法的“最小二乘”这个名字来自最小化平方欧几里得范数。

为了正式介绍这些选择和它们带来的解，状态-动作空间将被假设为有限的， $S = \{s_1, \dots, s_N\}$ ， $A = \{a_1, \dots, a_M\}$ 。然而，最终，3.3.3 节中实际的算法也可以应用于无限和连续的状态-动作空间。

一个线性参数化的 Q 函数可用如下形式表示：

$$\hat{Q}(s, a) = \sum_{l=1}^d \varphi_l(s, a) \theta_l = \phi^\top(s, a) \theta \quad (3.10)$$

其中， $\theta \in \mathbb{R}^d$ 是参数向量， $\phi(s, a) = [\varphi_1(s, a), \dots, \varphi_d(s, a)]^\top$ 是一个向量基函数 (BF)，也称为特征 [Bertsekas and Tsitsiklis, 1996][⊖]。因此，能被描绘的 Q 函数空间是 BF 的跨度， $\hat{\mathcal{Q}} = \{\phi^\top(\cdot, \cdot) \theta \mid \theta \in \mathbb{R}^d\}$ 。

给定一个加权函数 $\rho: S \times A \rightarrow [0, 1]$ ，Q 函数的 (平方) 加权欧几里得范数定义如下：

$$\|Q\|_\rho^2 = \sum_{\substack{i=1, \dots, N \\ j=1, \dots, M}} \rho(s_i, a_j) |Q(s_i, a_j)|^2$$

其中 $\|Q\|_\rho$ 是这个表达式的平方根，下文我们将主要使用方法的变体。

预计策略评估中的相应的加权最小二乘映射操作：

$$(\mathcal{Q}) = \arg \min \|\hat{\mathcal{Q}} - Q\|$$

权重函数可理解为状态-动作空间的概率分布，所以它们的加权和必须是 1。3.3.3 节中的无模型策略评估算法使用 ρ 的分布来生成样本。

1. 矩阵形式的贝尔曼映射

为了专门研究基于投影和 BRM 方法的线性近似 (欧几里得范数的情况)，我们将把贝尔曼映射写成一个矩阵形式，然后变成向量形式

81

因为状态-动作空间是离散的，贝尔曼映射可以写成如下形式：

$$[B_Q^\pi(Q)](s_i, a_j) = \sum_{i'=1}^N T(s_i, a_j, s_{i'}) [R(s_i, a_j, s_{i'}) + \gamma Q(s_{i'}, \pi(s_{i'}))] \quad (3.11)$$

$$= \sum_{i'=1}^N T(s_i, a_j, s_{i'}) R(s_i, a_j, s_{i'}) + \gamma \sum_{i'=1}^N T(s_i, a_j, s_{i'}) Q(s_{i'}, \pi(s_{i'})) \quad (3.12)$$

对于任意 i 和 j ，两部分求和可以表示如下：

$$B_Q^\pi(Q) = R + \gamma T^\pi Q \quad (3.13)$$

⊖ 本章使用列向量来表示特征。

其中 $B_Q^\pi: \mathbb{R}^{NM} \rightarrow \mathbb{R}^{NM}$ 。由 $[i, j] = i + (j-1)N$ 表示 i 和 j 相对应的向量[⊖]，向量和公式 (3.13) 右侧的矩阵定义如下[⊖]：

- $Q \in \mathbb{R}^{NM}$ 是 Q 函数的向量表示，有 $Q_{[i,j]} = Q(s_i, a_j)$ 。
- $R \in \mathbb{R}^{NM}$ 是回报函数 R 的期望的向量表示。其中元素 $R_{[i,j]}$ 是状态 s_i 采取动作 a_j 后的期望回报。例如 $R_{[i,j]} = \sum_{i'=1}^N T(s_i, a_j, s_{i'}) R(s_i, a_j, s_{i'})$ ，所以，针对公式 (3.12) 中的第一个和式，转换函数 T 已经被集成到了 R 中（不同于第二个和式）。
- $T^\pi \in \mathbb{R}^{NM \times NM}$ 是一个将策略和转换函数集成到一起的矩阵表示，其中 $T_{[i,j],[i',j']}^\pi = T(s_i, a_j, s_{i'})$ ，如果 $\pi(s_{i'}) = a_{j'}$ 或 0。关于 T^π ，一个有用的思考方式是它包含转换概率的状态-动作对，而不只是状态。在这样的解释中， $T_{[i,j],[i',j']}^\pi$ 是从任意状态-动作对 (s_i, a_j) 中转移到另外符合当前策略的状态-动作对 $(s_{i'}, a_{j'})$ 的概率，因此，如果 $a_{j'} \neq \pi(s_{i'})$ ，则概率如定义所说一样是 0。这个解释也表明，随机策略可以用一个简单的方法修改，在这种情况下， $T_{[i,j],[i',j']}^\pi = T(s_i, a_j, s_{i'}) \cdot \pi(s_{i'}, a_{j'})$ ，其中 $\pi(s, a)$ 是在 s 中选用 a 的概率。

下一步工作是用参数向量的形式复写贝尔曼映射。将公式 (3.13) 中的通用 Q 向量替换为近似的参数化 Q 向量。在所有方法中这是非常有用的，贝尔曼映射经常被用在近似 Q 函数中。使用下列矩阵表示 BF：

$$\Phi_{[i,j],d} = \varphi_d(s_i, a_j), \Phi \in \mathbb{R}^{NM \times d} \quad (3.14)$$

一个近似的 Q 向量被写作 $\hat{Q} = \Phi\theta$ 。将这个代入公式 (3.13) 可以得到：

$$B_Q^\pi(\Phi\theta) = R + \gamma T^\pi \Phi\theta \quad (3.15)$$

现在我们可以讨论线性情况下的预计策略评估和 BRM。需要注意的是公式 (3.15) 中的矩阵和向量太大了以至于无法直接在实际中使用。然而，我们将看到从这些大型矩阵和向量开始，预计策略评估和 BRM 的解都可以写成较小的矩阵和向量，可以在实际算法中进行存储和操作。

2. 预计策略评估

在 BF 的适当条件和权重 ρ 下（详情参见 3.6.1 节），线性情况下的映射贝尔曼估计可以用下列公式完全解决：

$$\hat{Q} = \Pi^\rho(B_Q^\pi(\hat{Q}))$$

以致 $\min_{\hat{Q} \in \mathcal{Q}} \|\hat{Q} - \Pi(B_Q^\pi(\hat{Q}))\|$ 的最小值为 0。在矩阵形式下，映射贝尔曼估计可以写作：

$$\begin{aligned} \Phi\theta &= \Pi^\rho B_Q^\pi(\Phi\theta) \\ &= \Pi^\rho(R + \gamma T^\pi \Phi\theta) \end{aligned} \quad (3.16)$$

其中 Π^ρ 是加权最小二乘投影 Π^ρ 的封闭形式的矩阵表示：

$$\Pi^\rho = \Phi(\Phi^\top \rho \Phi)^{-1} \Phi^\top \rho \quad (3.17)$$

权重矩阵 ρ 在其主对角线上收集了每一个状态-动作对的权重：

⊖ 如果 BF 向量的 d 元素被安排进一个 $N \times M$ 的矩阵，首先用前 N 个元素填写第一列，然后用随后的 N 个元素填写第二列，等等。位于索引 $[i, j]$ 的元素将被放在矩阵的第 i 行和第 j 列。

⊖ 函数和映射的向量或矩阵用黑体字表示，普通向量和矩阵则用正常的字体显示。

$$\rho_{[i,j][i,j]} = \rho(s_i, a_j), \rho \in \mathbb{R}^{NM \times NM} \quad (3.18)$$

将公式 (3.17) 带入公式 (3.16) 后, 经过一个 $\Phi^T \rho$ 的左乘和整理, 我们可以得到:

$$\Phi^T \rho \Phi \theta = \gamma \Phi^T \rho T^\pi \Phi \theta + \Phi^T \rho R$$

或用一种压缩的表达方式:

$$A\theta = \gamma B\theta + b \quad (3.19)$$

其中 $A = \Phi^T \rho \Phi$, $B = \Phi^T \rho T^\pi \Phi$, $b = \Phi^T \rho R$ 矩阵 A 和 B 是 $\mathbb{R}^{d \times d}$ 的矩阵, 其中 b 是 \mathbb{R}^d 的一个向量。这是一个重要的表达, 强调了投影贝尔曼方程可以被小的矩阵和向量 (大小为 $d \times d$ 和 d) 表示和解决, 而不是最初在公式中出现的大向量和矩阵 (大约有 $NM \times NM$ 的大小)。

83

接下来, 介绍了两个用于预计策略评估的理想化算法。假定具有 A 、 B 和 b 的知识。下一部分将演示如何去掉这个假设。

理想 LSTD-Q 算法属于图 3.1 中预计策略评估图中单次迭代方法的算法子类。它简单地解决了系统公式 (3.19) 去达到参数向量 θ 。该参数向量提供所考虑策略 π 的近似 Q 函数 $\hat{Q}^\pi(s, a) = \phi^\top(s, a)\theta$, 需要注意的是因为 θ 在公式 (3.19) 的两侧都出现了, 这个公式可以简化为:

$$(A - \gamma B)\theta = b$$

理想化的 LSPE-Q 是一个迭代算法, 因此属于图 3.1 预计策略评估方法中的第二个子类。同时它也依赖于公式 (3.19), 但是更新了参数向量的增量:

$$\theta_{t+1} = \theta_t + \alpha(\theta'_{t+1} - \theta_t) \quad (3.20)$$

其中

$$A\theta'_{t+1} = \gamma B\theta_t + b$$

从初值 θ_0 开始, 在这个更新中, α 是一个正的步长参数。

3. 贝尔曼残差最小化

用 BRM-Q 公式 (3.9) 求解, 本节中我们使用如下加权欧几里得范数:

$$\min_{\hat{Q} \in \mathcal{Q}} \left\| \hat{Q} - B_Q^\pi(\hat{Q}) \right\|_\rho^2 \quad (3.21)$$

使用 B^π 的矩阵表达式公式 (3.15), 可以用下列参数向量来重写这个最小化问题:

$$\begin{aligned} \min_{\theta \in \mathbb{R}^d} \left\| \Phi\theta - R - \gamma T^\pi \Phi\theta \right\|_\rho^2 \\ = \min_{\theta \in \mathbb{R}^d} \left\| (\Phi - \gamma T^\pi \Phi)\theta - R \right\|_\rho^2 \\ = \min_{\theta \in \mathbb{R}^d} \left\| C\theta - R \right\|_\rho^2 \end{aligned}$$

其中 $C = \Phi - \gamma T^\pi \Phi$ 。可以用下列公式来求解这个问题的最小值:

$$C^\top \rho C \theta = C^\top \rho R \quad (3.22)$$

理想化的 BRM-Q 算法包括求解该方程以得到 θ , 并且因此得到所考虑的策略的近似 Q 函数。

84

值得注意的是, BRM 与预计策略评估密切相关, 因为它可以解释为解类似的投影方程 [Bertsekas, 2011a]。具体来说, 在公式 (3.21) 中找到最小值等价于求解:

$$\Phi\theta = \Pi^\pi(\bar{B}_Q^\pi(\Phi\theta))$$

除了改进了的映射 \bar{B}_Q^π , 与投影贝尔曼方程 (公式 (3.16)) 相同:

$$\bar{B}_Q^\pi(\Phi\theta) = B_Q^\pi(\Phi\theta) + \gamma(T^\pi)^\top[\Phi\theta - B_Q^\pi(\Phi\theta)]$$

3.3.3 无模型算法的实现

为了获得出现在它们的方程中的矩阵和向量，上面给出的理想化算法需要马尔可夫决策过程的转移和回报函数，这在强化学习中是未知的。此外，算法需要搜索所有的状态-动作对，这在大规模状态-动作空间中是不可能的。

这意味着，在实际强化学习中，应该使用这些算法的基于样本的版本。幸运的是，由于涉及的矩阵和向量的特殊结构，它们可以从样本中估计得到。在本节，我们将导出可实现的 LSTD-Q、LSPE-Q 和 BRM-Q 算法。

1. 预计策略评估

考虑具有形如 (s_i, a_i, s'_i, r_i) , $i = 1, \dots, n$ 的一组 n 个转换样本，其中 s'_i 是从分布 $T(s_i, a_i, \cdot)$ 和 $r_i = R(s_i, a_i, s'_i)$ 中得出。状态-动作对 (s_i, a_i) 必须从由权重函数 ϕ 给出的分布得出。例如，当生成模型可用时，可以独立于 ρ 地从样本中抽取，并且该模型可以用于找到对应的下一状态以及回报。从相反的角度来看，也可以说状态-动作对的分布意味着其在投影贝尔曼方程中使用的权重。例如，沿着系统的一组轨迹收集样本，或沿着单个轨迹收集样本，隐式地导出分布（权重） ρ 。

在公式 (3.19) 和公式 (3.20) 中出现的矩阵 A 和 B 以及向量 b 的估计可以从样本中构建如下：

$$\begin{aligned}\hat{A}_i &= \hat{A}_{i-1} + \phi(s_i, a_i) \phi^\top(s_i, a_i) \\ \hat{B}_i &= \hat{B}_{i-1} + \phi(s_i, a_i) \phi^\top(s'_i, \pi(s'_i)) \\ \hat{b}_i &= \hat{b}_{i-1} + \phi(s_i, a_i) r_i\end{aligned}\quad (3.23)$$

从初始值为 0 开始 ($\hat{A}_0 = 0, \hat{B}_0 = 0, \hat{b}_0 = 0$)。

LSTD-Q 使用公式 (3.23) 处理 n 个样本，然后解方程：

$$\frac{1}{n} \hat{A}_n \theta = \gamma \frac{1}{n} \hat{B}_n \theta + \frac{1}{n} \hat{b}_n \quad (3.24)$$

或者等价于：

$$\frac{1}{n} (\hat{A}_n - \gamma \hat{B}_n) \theta = \frac{1}{n} \hat{b}_n$$

为了找到参数向量 θ 。注意，因为这个方程是公式 (3.19) 的近似解，参数向量 θ 仅是公式 (3.19) 解的近似，(但是，为了简单起见，我们以相同的方式表示)。除以 n 可以防止系数随着处理更多样本而变得太大，进而增加算法的数值稳定性。复合矩阵 $(\hat{A} - \gamma \hat{B})$ 也可以被更新为单个实体 $(\hat{A} - \gamma \hat{B})$ ，从而消除了在存储器中存储两个很大的矩阵 \hat{A} 和 \hat{B} 的需求。

算法 7 总结了这种存储方面更高效的 LSTD-Q 的算法变体。注意 $(\hat{A} - \gamma \hat{B})$ 的更新简单地累加了公式 (3.23) 中的 \hat{A} 和 \hat{B} ，其中 \hat{B} 被乘以适当的 $-\gamma$ 。

```

1: input policy to evaluate  $\pi$ , BF's  $\phi$ , samples  $(s_i, a_i, s'_i, r_i)$ ,  $i = 1, \dots, n$ 
2:  $(\hat{A} - \gamma \hat{B})_0 \leftarrow 0, \hat{b}_0 \leftarrow 0$ 
3: for  $i = 1, \dots, n$  do
4:    $(\hat{A} - \gamma \hat{B})_i \leftarrow (\hat{A} - \gamma \hat{B})_{i-1} + \phi(s_i, a_i) \phi^\top(s_i, a_i) - \gamma \phi(s_i, a_i) \phi^\top(s'_i, \pi(s'_i))$ 
5:    $\hat{b}_i \leftarrow \hat{b}_{i-1} + \phi(s_i, a_i) r_i$ 
6: solve  $\frac{1}{n} (\hat{A} - \gamma \hat{B})_n \theta = \frac{1}{n} \hat{b}_n$ 
7: output Q-function  $\hat{Q}^\pi(s, a) = \phi^\top(s, a) \theta$ 

```

算法 7 LSTD-Q

将 LSTD-Q 策略评估与精确的策略改进结合起来可能是最小二乘类中最广泛使用的策略迭代算法, 简称最小二乘策略迭代 (LSPI)。

LSPE-Q 使用与 LSTD-Q 相同的方式估计 \hat{A} 、 \hat{B} 、 \hat{b} , 但算法迭代地更新参数向量。在其基本变体中, LSPE-Q 以任意初始值开始迭代地求参数向量 θ_0 , 并使用:

$$\theta_i = \theta_{i-1} + \alpha(\theta'_i - \theta_{i-1})$$

$$\text{其中 } \frac{1}{i}\hat{A}_i\theta'_i = \gamma\frac{1}{i}\hat{B}_i\theta_{i-1} + \frac{1}{i}\hat{b}_i \quad (3.25)$$

此更新是理想化版本公式 (3.20) 的近似的基于样本的版本。与 LSTD-Q 类似, 除以 i 增加了更新的数值稳定性。由于此时仅仅处理了几个样本, 估计 \hat{A} 在学习过程的开始时是不可逆的。这个问题的比较实用的解决方案是用单位矩阵的小倍数去初始化 \hat{A} 。

通过 1) 在参数向量的连续更新过程中处理多于一个样本, 2) 在处理每个 (每批) 参数之后更新多于一个参数, 可以获得比公式 (3.25) 更灵活的算法, 同时保持系数估计值不变。前者的修改可以增加算法的稳定性, 特别是在学习的早期阶段, 而后者可以加速其收敛, 特别是在后期阶段, 因为对 \hat{A} 、 \hat{B} 、 \hat{b} 的估计变得更精确。

算法 8 展示了 LSPE-Q 的这种更灵活的变体, 其中第一步是更新 \bar{n} 个样本的中间参数 (\bar{n} 最好是 n 的倍数)。在每次处理中, 参数向量更新 N_{upd} 次。需要注意, 与公式 (3.25) 不同的是, 参数更新索引 τ 不同于样本索引 i , 因为这两个索引不再同步地更新。

```

1: input policy to evaluate  $\pi$ , BF's  $\phi$ , samples  $(s_i, a_i, s'_i, r_i)$ ,  $i = 1, \dots, n$ 
   step size  $\alpha$ , small constant  $\delta_A > 0$ 
   batch length  $\bar{n}$ , number of consecutive parameter updates  $N_{\text{upd}}$ 
2:  $\hat{A}_0 \leftarrow \delta_A I$ ,  $\hat{B}_0 \leftarrow 0$ ,  $\hat{b}_0 \leftarrow 0$ 
3:  $\tau = 0$ 
4: for  $i = 1, \dots, n$  do
5:    $\hat{A}_i \leftarrow \hat{A}_{i-1} + \phi(s_i, a_i)\phi^\top(s_i, a_i)$ 
6:    $\hat{B}_i \leftarrow \hat{B}_{i-1} + \phi(s_i, a_i)\phi^\top(s'_i, \pi(s'_i))$ 
7:    $\hat{b}_i \leftarrow \hat{b}_{i-1} + \phi(s_i, a_i)r_i$ 
8:   if  $i$  is a multiple of  $\bar{n}$  then
9:     for  $\tau = \tau, \dots, \tau + N_{\text{upd}} - 1$  do
10:       $\theta_{\tau+1} \leftarrow \theta_\tau + \alpha(\theta'_\tau - \theta_\tau)$ , where  $\frac{1}{i}\hat{A}_i\theta'_\tau = \gamma\frac{1}{i}\hat{B}_i\theta_\tau + \frac{1}{i}\hat{b}_i$ 
11: output Q-function  $\hat{Q}^\pi(s, a) = \phi^\top(s, a)\theta_{\tau+1}$ 

```

算法 8 LSPE-Q

由于 LSPE-Q 必须对公式 (3.25) 中的系统进行多次求解, 因此比 LSTD-Q 需要更多的计算量, LIST-Q 求解公式 (3.24) 的系统只需一次类似的计算。另一方面, LSPE-Q 的增量式处理的特性可以提供好于 LSTD-Q 的优势。例如, LSPE 可以受益于参数向量的良好初始值, 并且可以通过控制步长来实现更好的灵活性。

2. 贝尔曼残差最小化

接下来将简要讨论基于样本的 BRM 算法。出现在理想化 BRM 方程 (公式 (3.22)) 中的矩阵 $C^\top \rho C$ 和向量 $C^\top \rho R$ 可通过样本估计得到。估计过程需要双转换样本, 也就是说, 对于每个状态-动作样本, 两个独立状态下的转换必须是可用的。这些双转换样本有如下形式 $(s_i, a_i, s'_{i,1}, r_{i,1}, s'_{i,2}, r_{i,2})$, $i = 1, \dots, n$, 其中 $s'_{i,1}$ 和 $s'_{i,2}$ 都是 $T(s_i, a_i, \cdot)$ 的独立分布, 其中 $r_{i,1} = R(s_i, a_i, s'_{i,1})$ 且 $r_{i,2} = R(s_i, a_i, s'_{i,2})$ 。使用这些样本, 可以构建如下估计:

$$\begin{aligned}
\widehat{(C^T \rho C)}_i &= \widehat{(C^T \rho C)}_{i-1} + \\
&\quad \left[\phi(s_i, a_i) - \gamma \phi(s'_{i,1}, \pi(s'_{i,1})) \right] \cdot \left[\phi^T(s_i, a_i) - \gamma \phi^T(s'_{i,2}, \pi(s'_{i,2})) \right] \\
\widehat{(C^T \rho R)}_i &= \widehat{(C^T \rho R)}_{i-1} + \phi(s_i, a_i) - \gamma \phi(s'_{i,2}, \pi(s'_{i,2}))
\end{aligned} \tag{3.26}$$

从初始值 0 开始: $\widehat{(C^T \rho C)}_0 = 0$, $\widehat{(C^T \rho R)}_0 = 0$ 。一旦处理了所有的样本, 上式可以近似求解公式 (3.22), 获得一个参数向量, 从而近似地求解 Q 函数。

使用双样本的原因是, 如果一个样本 (s_i, a_i, s'_i, r_i) 是用于构建形如 $[\phi(s_i, a_i) - \gamma \phi(s'_i, \pi(s'_i))] \cdot [\phi^T(s_i, a_i) - \gamma \phi^T(s'_i, \pi(s'_i))]$ 的样本。这些样本将是有偏差的, 这反过来会导致对 $C^T \rho C$ 的估计出现偏差, 并让算法变得不那么可靠 [Baird, 1995; Bertsekas, 1995]。

3. 探索需求

上述所有算法中的一个关键问题是“探索”: 确保状态-动作空间被可用样本充分覆盖。首先考虑对动作空间的探索。算法通常用于评估一个确定的策略。如果仅根据当前策略 π 收集样本, 即如果所有样本具有形式 $(s, \pi(s))$, 则不能获得 $a \neq \pi(s)$ 时的 (s, a) 对信息。因此, 这样的近似 Q 值无法被充分估计, 对策略的改进也不可靠。为了解决这个问题, 探索是必要的: 有时, 必须选择不同于 $\pi(s)$ 的动作, 例如以随机方式。现在来看状态空间, 当沿着系统的轨迹收集样本时, 探索也将起着作用。在缺乏探索的情况下, 在当前策略下未被访问的状态空间的区域将不会在样本集中表示, 导致在这些区域中的价值函数估计效果不佳, 即使它们可能对求解问题很重要。相反地, 探索驱动着系统沿状态空间的更大区域去求解。

4. 计算方面的注意事项

用于策略评估的最小二乘算法的计算开销主要是求解出现在所有算法中的线性系统。然而, 对于诸如 LSTD 和 BRM 这样的单次迭代算法 (其仅需要求解系统一次), 如果样本的数量非常大, 则该算法主要花时间在处理样本上。

线性系统可以通过几种方式求解, 例如矩阵求逆高斯消去, 或用 Sherman-Morrison 公式递增地计算倒数 (参见 [Golub and Van Loan, 1996]。还要注意的, 当 BF 向量是稀疏的时候, 如在局部 BF 经常遇到的情况中, 可以利用该稀疏度来极大地提高所有最小二乘算法中的矩阵和向量更新的计算效率。

3.3.4 参考文献

3.3.1 节介绍 [Farahmand et al, 2009] 的研究。之后, 在第 3 章中介绍 [Bus oniu et al, 2010a] 推导。

LSTD 是在 [Bradtke and Barto, 1996] 的上下文中介绍 V 函数 (LSTD-V), 并且理论由 [Boyan, 2002; Konda, 2002]; Nedić and Bertsekas, 2003; Lazaric et al, 2010b; Yu, 2010] 研究。LSTD-Q (Q 函数情况的扩展) 由 [Lagoudakis et al, 2002] 引入; [Lagoudakis and Parr, 2003a] 也使用它来开发 LSPI 算法。LSTD-Q 有各种使用和扩展方式, 例如 [Xu et al, 2007; Li et al, 2009; Kolter and Ng, 2009; Buşoniu et al, 2010d, b; Thiery and Scherrer, 2010]。

LSPE-V 由 [Bertsekas and Ioffe, 1996] 引入并由 [Nedić and Bertsekas, 2003; Bertsekas et al, 2004; Yu and Bertsekas, 2009] 进行理论研究。它对 Q 函数 (LSPE-Q) 的改进被 [Jung and Polani, 2007a, b; Buşoniu et al, 2010d] 采用。

贝尔曼残差最小化的想法是由 [Schweitzer and Seidmann, 1985] 最早提出。[Lagoudakis and Parr, 2003a; Antos et al, 2008; Farahmand et al, 2009] 研究了 BRM-Q 及其变体；[Scherrer, 2010] 比较了 BRM 方法和预测方法。[Antos et al, 2008; Farahmand et al, 2009] 的研究变体被 [Farahmand et al, 2009] 称为“改性 BRM”，其通过引入最小化问题的变化（公式（3.9））来消除双重采样的必要性。

3.4 策略迭代的在线最小二乘法

在实际的应用中的一个重要问题是将最小二乘法应用于在线学习中。不同于只重视最终性能的离线学习，在线学习中，性能应该每隔几个转换样本进行一次改进。通过在每 n 个转换样本上进行一次策略改进，策略迭代可以实现准确的评估。这种变体称为乐观策略（optimistic policy）迭代 [Bertsekas and Tsitsiklis, 1996; Sutton, 1988; Tsitsiklis, 2002]。在极端情况下（完全乐观的情况下），每次转换样本都会导致策略改进。乐观的策略更新与 LSTD-Q 结合，从而获得乐观（optimistic）的 LSPI [Busoniu et al, 2010d, b] 和 LSPE-Q [Jung and Polani, 2007a, b]。[Li et al, 2009] 探索了一种非乐观的、计算量更大的在线策略迭代方法，其中 LSPI 在连续采集样本集之间完全地探索。

89

```

1: input BF $\phi$ , policy improvement interval  $L$ , exploration schedule,
   initial policy  $\pi_0$ , small constant  $\delta_A > 0$ 
2:  $k \leftarrow 0, t \leftarrow 0$ 
3:  $(A - \gamma B)_0 \leftarrow \delta_A I, \hat{b}_0 \leftarrow 0$ 
4: observe initial state  $s_0$ 
5: repeat
6:    $a_t \leftarrow \pi_k(s_t) + \text{exploration}$ 
7:   apply  $a_t$ , observe next state  $s_{t+1}$  and reward  $r_{t+1}$ 
8:    $(A - \gamma B)_{t+1} \leftarrow (A - \gamma B)_t + \phi(s_t, a_t)\phi^\top(s_t, a_t) - \gamma\phi(s_t, a_t)\phi^\top(s_{t+1}, \pi(s_{t+1}))$ 
9:    $\hat{b}_{t+1} \leftarrow \hat{b}_t + \phi(s_t, a_t)r_{t+1}$ 
10:  if  $t = (k+1)L$  then
11:    solve  $\frac{1}{t}(A - \gamma B)_{t+1}\theta_k = \frac{1}{t}\hat{b}_{t+1}$  to find  $\theta_k$ 
12:     $\pi_{k+1}(s) \leftarrow \arg \max_{a \in A} \phi^\top(s, a)\theta_k \quad \forall s$ 
13:     $k \leftarrow k+1$ 
14:     $t \leftarrow t+1$ 
15: until experiment finished

```

算法 9 在线 LSPI

接下来，我们简要讨论在 [Busoniu et al, 2010d] 中提出的在线乐观 LSPI 方法，如算法 9 所示。采用了与离线 LSTD-Q 和 LSPI 相同的矩阵和向量估计，但有重要的区别。首先，在线 LSPI 通过使用其当前策略来收集其自己的样本并与系统交互。这即意味着必须在策略上进行明确的探索。其次，执行乐观的策略改进，即改进策略而不等待估计 $(A - \gamma B)$ 和 \hat{b} 接近于它们对于当前策略的渐近值。此外，继续更新这些估计值而不是在策略发生变化之后才重置，因此，实际上它们对应于多个策略。这里假设 $A - \gamma B$ 和 b 对于后续策略是相似的。这种高计算成本的替代算法存储样本并在每个策略更新之前从头开始估计，但在 3.5 节中将会证明这在实践中是不必要的。

连续策略改进之间的转换数 L 是算法的关键参数。例如，当 $L = 1$ 时，在线 LSPI 是完全乐观的。通常， L 不应该太大，以避免不良的策略被使用太久。需要注意的是，与在离线情况下一样，改进的策略不必在线 LSPI 中显式地计算，而是可以根据需要进行计算。

90

3.5 例子: car-on-the-hill

为了举例说明策略迭代的最小二乘法的作用, 我们将两种这样的方法 (离线 LSPI 和在线乐观 LSPI) 应用到 “car-on-the-hill” 问题 [Moore and Atkeson, 1995], 这是一个近似强化学习的经典基准问题 (benchmark)。由于其维度较低, 这个问题可以使用简单的线性逼近器解决, 其中 BF 被分布在等距网格中。这使我们不必费力地去定制 BF 的困难或者使用非参数近似方法, 而是专注于算法的动作。

在 “car-on-the-hill” 的问题中, 一个质点 (汽车) 必须用一个水平的力驱动经过无摩擦的山顶, 见图 3.2a。对于一些初始状态, 由于可用力是有限的, 汽车必须首先沿着相反的坡度驱动到左边, 并且在向右加速之前获得朝向目标的动量。

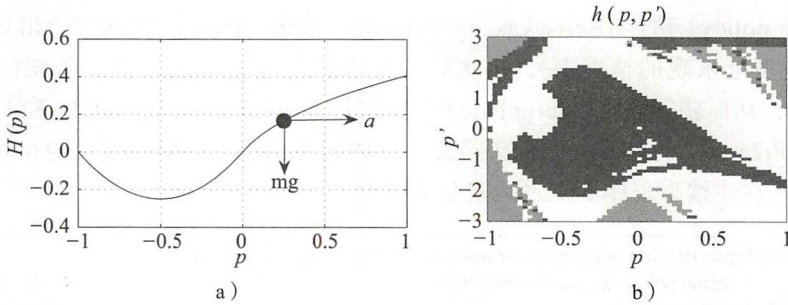


图 3.2 a) “car-on-the-hill” 中汽车显示为黑色点; b) 次最优策略 (黑色表示 $a = -4$, 白色表示 $a = +4$, 灰色表示两个动作效果相同)

p 表示汽车的水平位置, 其动力学方程是 [Ernst et al, 2005] 的变体:

$$\ddot{p} = \frac{a - 9.81H'(p) - \dot{p}^2 H''(p)}{1 + [H'(p)]^2} \quad (3.27)$$

其中山形 $H(p)$ 由 $p^2 + p$ 给出, 其中 $p < 0$; $p \geq 0$ 时等于 $p^{-1}\sqrt{1+5p^2}$ 。符号 \dot{p} 和 \ddot{p} 表示 p 的第一和第二时间导数, 而 $H'(p)$ 和 $H''(p)$ 表示 H 对 p 的一阶和二阶导数。状态变量是位置和速度 $s = [p, \dot{p}]^T$, 动作 a 是施加的力。离散时间动态 T 通过使用 0.1 秒的离散时间步长在连续时间步长之间进行数值积分 (公式 (3.27)) 获得。因此, s_t 和 p_t 是连续变量 s 和 p 的采样值。状态空间是 $S = [-1, 1] \times [-3, 3]$ 加上 s_{t+1} 落在这些边界之外时达到的终端状态, 离散动作空间是 $A = \{-4, 4\}$ 。目标是在允许的极限内的速度向右驶过山的顶部, 同时以任何其他方式到达终点状态被认为是失败。为了表达这一目标, 选择以下回报函数:

$$R(s_t, a_t, s_{t+1}) = \begin{cases} -1 & , p_{t+1} < -1 \text{ 或 } |\dot{p}_{t+1}| > 3 \\ 1 & , p_{t+1} > 1 \text{ 且 } |\dot{p}_{t+1}| \leq 3 \\ 0 & \text{其他} \end{cases} \quad (3.28)$$

阻尼系数 $\gamma = 0.95$ 。图 3.2b 显示了这种回报函数的次最优策略。

为了应用上述算法, Q 函数状态上近似空间在等距的 13×13 网格上使用双线性插值来逼近。为了在离散动作空间上表示 Q 函数, 为两个离散动作存储单独的参数, 因此逼近器可以写为:

$$\hat{Q} = (s, a_j) = \sum_{i=1}^{169} \varphi_i(s) \theta_{i,j}$$

其中由状态相关的BF $\varphi_i(s)$ 提供内插系数, 对于任何 s , 至多 4 个 BF 为非零, 并且 $j = 1, 2$ 。通过为两个离散动作复制状态相关的 BF, 该逼近器可以以标准形式 (公式 (3.10)) 写入, 并且可以用于策略迭代的最小二乘法。

首先, 我们将使用这个逼近器的 LSPI 用于 “car-on-the-hill” 问题。独立地生成一组 10 000 个随机的、均匀分布的状态-动作样本; 这些样本重复用在每个策略迭代时对策略进行评估。使用这些设置, LSPI 通常在 7 到 9 次迭代后就收敛 (20 次独立运行, 其中当连续参数向量之间的差异下降到 0.001 以下时, 实现收敛)。图 3.3a 显示了在代表性运行期间发现的策略的后果。根据所选择的表示的分辨率限制, 找到了图 3.2 的次最优策略的合理近似。

为了比较, 我们还使用相同的近似和相同的收敛阈值运行近似值迭代算法。(实际的算法, 称为模糊 Q 迭代, 在 [Busoniu et al, 2010c] 中描述; 这里, 我们只关注它是近似值迭代)。算法在 45 次迭代之后收敛。在实践中经常观察到值迭代与策略迭代相比收敛得较慢。图 3.3b 显示了通过值迭代找到的策略的子序列。类似于我们考虑的 PI 算法, 策略由近似 Q 函数隐式地表示, 特别是通过最大化 Q 函数来选择动作, 如在公式 (3.5) 中。最终的解决方案不同于 LSPI 发现的解决方案, 并且算法也有不同的收敛: LSPI 最初在每次迭代时在策略空间中产生大的步长 (例如, 策略的结构在第二次迭代之后已经可见), 而值迭代则进行小步的、增量式的迭代。

92

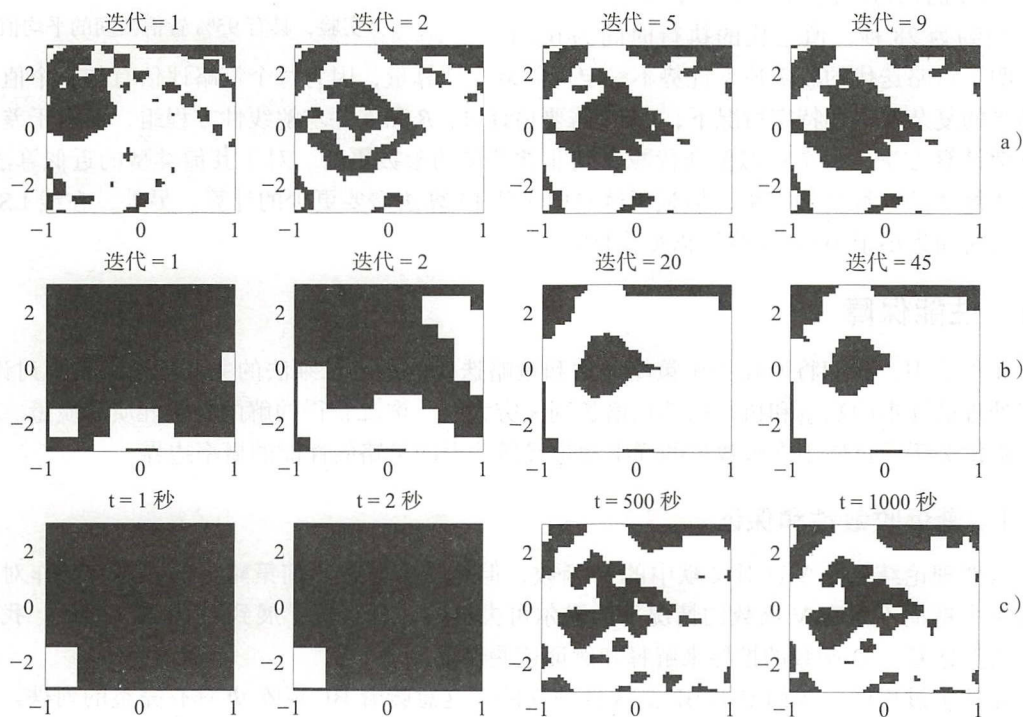


图 3.3 由上述算法产生的子序列的代表策略。a) 离线 LSPI; b) 模糊 Q-迭代; c) 在线 LSPI。对于轴和颜色的意义, 请参见图 3.2b, 另外注意, 当两个动作结果相同时, 优先选择负 (黑色) 动作

最后, 将在线、乐观的 LSPI 算法应用于“car-on-the-hill”问题中。该实验模拟运行 1000 秒的时间, 已经收集了 10 000 个样本, 类似于离线算法的做法。该间隔被分成单独的学习试验, 在随机初始状态下初始化, 并且当达到终端状态时停止, 否则在 3 秒之后停止。每 10 个样本 (即每 1 秒) 执行一次策略改进: 其间使用 ε -贪婪探索策略 (见第 1 章), ε 初始为 1, 并且呈指数形式衰减, 使其在 350 秒之后达到 0.1。图 3.3c 显示了在运行期间找到的策略的子序列。在策略空间中在线 LSPI 比离线 LSPI 做更小的步骤, 因为在连续的策略改进之间, 它处理其来自状态空间的较小区域中的更少的样本。事实上, 在学习结束时, 在线 LSPI 已经处理了每个样本一遍, 而离线 LSPI 在每次迭代时都处理所有样本一遍。

图 3.4 显示了在线学习过程与离线 LSPI 得到的最终策略的性能。性能通过初始状态的等距网格上的平均经验回报来测量, 模拟的评估精度为 0.001, 我们称之为平均回报“得分”。尽管理论上并不确定它是否收敛且次优, 在线 LSPI 找到的解至少与离线算法一样好 (令人鼓舞的是还在其他几个问题上有应用, 详情参见 [Buşoniu et al, 2010d, b; Buşoniu et al, 2010a])。为了完整性, 我们还报告了值迭代解的分数为 0.219, 略低于由任一版本的 LSPI 获得的分数。

LSPI 的执行时间约为 34 秒, 在线 LSPI 的执行时间约为 28 秒, 值迭代的执行时间为 0.3 秒。

这说明了策略迭代的收敛速率优势不一定可以减少计算量, 因为每个策略评估有与整个值迭代相当的复杂度。在特定情况下, LSPI 需要估计 A 、 B 和 b 并求解线性方程组, 而对于差值的近似计算方法, 每个近似值迭代减少到非常简单的参数更新。对于其他类型的近似算法, 值迭代算法的计算更加密集, 但是仍然趋向于比 PI 算法需要更少的计算。另外, 在线 LSPI 的执行时间远小于 1000 秒的模拟实验时间。

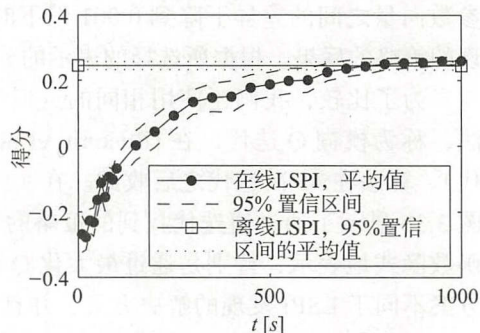


图 3.4 与离线 LSPI 相比, 在线 LSPI 中的策略得分的演变。记录来自 20 个独立实验, 具有 95% 置信区间的平均值

3.6 性能保障

在本节中, 我们将回顾关于策略评估和策略迭代的最小二乘法的主要理论。首先讨论, 随着处理的样本的数量和执行的迭代增长到无穷大时, 渐近获得的解的收敛性质和质量。然后, 提供关于通过使用有限数量的样本和迭代所获得的策略的性能的概率边界。

3.6.1 渐近收敛性和保证

虽然理论结果主要针对文献中的 V 函数, 但是当考虑在当前策略下的状态-动作对的马尔可夫链而不是在 V 函数的情况下的马尔可夫链时, 可直接扩展到 Q 函数。因此, 我们将使用上述基于 Q 函数的推导来解释和例证这些保证。

在整个过程中, 我们要求 BF 是线性独立的, 这意味着 BF 矩阵 Φ 具有完整的列秩。直观地说, 这意味着没有冗余 BF 。

1. 预计策略评估

在预计策略评估的背景下, LSTD 类算法和 LSPE 类算法之间有如下的一个重要区别: 每当公式 (3.19) 有解时, LSTD- Q 算法将产生一个有意义的解决方案和许多权重函数 ρ 。

相反, 为了保证 LSPE-Q 算法迭代的收敛, 通常必须另外要求采样遵循 ρ^π , 由所考虑的策略 (简称为 π 的平稳分布) 引起的状态-动作对的平稳分布。直观地说, 这意味着每个状态-动作对 (s, a) 的权重等于该对沿用策略 π 生成的无限长轨迹的稳态概率。投影映射 Π^{ρ^π} 是关于由平稳分布 ρ^π 加权的非扩展的范数 $\|\cdot\|_{\rho^\pi}$, 并且因为原始贝尔曼映射 B_Q^π 是相对于该范数的缩写, 投影贝尔曼映射 $\Pi^{\rho^\pi}(B_Q^\pi(\cdot))$ 也是缩写^①。

需要确定的是 LSTD 不是非常依赖于使用 ρ^π , Yu (2010) 证明了通过微小的修改, LSTD 找到的解收敛为 $n \rightarrow \infty$, 即使一个策略使用来自不同策略的样本进行评估, 如所谓的非策略案例。即使对于 LSPE, 也可以通过控制步长 α 来减轻违反收敛假设的不稳定效应。此外, 有人已经提出了改进的类似 LSPE 的更新算法, 其收敛不要求 $\Pi_{\rho}(B_Q^\pi(\cdot))$ 是一个收缩, 参见 Bertsekas (2011a, b)。这些类型的结果在实践中是重要的, 因为它们为重用样本来评估不同策略 (即, 在整个 PI 算法的不同迭代次数) 做好了准备。相反, 如果必须遵循稳定分布, 则必须在每次迭代时使用当前策略生成新的样本。

95

为了简单起见, 现在假设使用平稳分布 ρ^π , 并且因此投影贝尔曼方程 (公式 (3.19)) 具有唯一解 (投影贝尔曼算子是一个压缩 (contraction) 并且允许一个唯一的固定点), 以下非正式的直观的推理有助于理解基于样本的 LSTD-Q 和 LSPE-Q 算法的收敛。渐近地, 如 $n \rightarrow \infty$, 可以确定 $\frac{1}{n}\hat{A}_n \rightarrow A$ 、 $\frac{1}{n}\hat{B}_n \rightarrow B$ 和 $\frac{1}{n}\hat{b}_n \rightarrow b$, 因为状态-动作样本的经验分布收敛到 ρ , 然而对于每一个 (s, a) 对的下一个状态样本 s' 的经验分布都收敛到 $T(s, a, s')$ 。因此, 实际的基于样本的 LSTD-Q 算法收敛到其理想化版本 (公式 (3.19)), 并且 LSTD-Q 可以渐近地找到投影贝尔曼方程的解。类似地, 基于样本的 LSPE-Q 渐近地变为等于它的理想化版本 (公式 (3.20)), 其仅是公式 (3.19) 的增量变体, 并且因此将产生相同的解决方案。事实上, 可以另外表明, 随着 n 的增长, LSTD-Q 和 LSPE-Q 的解彼此收敛得比收敛到它们的极限的速度更快, 参见 [Yu and Bertsekas, 2009]。

现在让我们来检验解决方案的准确性。在平稳分布 ρ^π , 我们有 [Bertsekas, 2011a; Tsitsiklis and Van Roy, 1997]:

$$\|Q^\pi - \hat{Q}^\pi\|_{\rho^\pi} \leq \frac{1}{\sqrt{1-\gamma^2}} \|Q^\pi - \Pi^{\rho^\pi}(Q^\pi)\|_{\rho^\pi} \quad (3.29)$$

其中 \hat{Q}^π 是由参数 θ 给出的 Q 函数, 其求解了 $\rho = \rho^\pi$ 的投影贝尔曼方程 (公式 (3.19))。因此, 我们用真实 Q 函数 Q^π 及其投影 $\Pi^{\rho^\pi}(Q^\pi)$ 之间的距离 $\|Q^\pi - \Pi^{\rho^\pi}(Q^\pi)\|_{\rho^\pi}$ 描述逼近器的代表能力。随着逼近器变得更强大, 该距离逐渐减小。然后, 预计策略评估导致具有与该距离成比例的误差的近似 Q 函数 \hat{Q}^π 。该比例由阻尼系数 γ 给出, 并且随着 γ 接近 1 而增长。最近已经有研究人员从 Q 函数的 \hat{Q} 的设置来优化这个工作 [Scherrer, 2010; Yu and Bertsekas, 2010]。

2. 贝尔曼残差最小化

以下关系适用于任何 Q 函数 Q , 例如参见 [Scherrer, 2010]

① 应用于函数 f 的投影映射 Π^{ρ^π} 。空间 \mathcal{F} 将 \mathcal{F} 中的最接近的元素返回到函数 f , 其中根据 L2 范数和测量 ρ^π 来定义距离。

② 如果对于任何 x, x' , $\|f(x) - f(x')\| \leq \gamma \|x - x'\|$, 映射 $f(x)$ 是一个压缩 (其中 $\gamma < 1$)。如果不等式适用于 $\gamma = 1$, 则映射是非扩展的 (较弱的属性)。

$$\|Q^\pi - Q\|_{\rho^\pi} \leq \frac{1}{1-\gamma} \|Q - B_Q^\pi(Q)\|_{\rho^\pi} \quad (3.30)$$

其中 ρ^π 是平稳分布。现在考虑策略上的 BRM 解决方案, Q 函数 \hat{Q}^π 的参数由 $\rho = \rho^\pi$ 时 BRM 方程 (公式 (3.22)) 的解给出。因为这个 Q 函数使公式 (3.30) 的右侧最小化, 所以解的误差 $\|Q^\pi - \hat{Q}^\pi\|_{\rho^\pi}$ 也小。

将预计策略评估与 BRM 相比较, 不能对其解决方案的相对质量做出普适性结论。例如, 只在策略评估的背景下, [Scherrer, 2010] 提出了一个基于实证的研究, 相比其他方法, 预计策略评估更多地胜过 BRM, 但是当它失败时, 它可能会比 BRM 的错误大得多。对于其他的结果和比较, 详见 [Munos, 2003; Scherrer, 2010]。

3. 近似策略迭代

关于策略迭代的一般结果可以根据如下无穷大范数给出。如果在每次 $k \geq 0$ 的迭代时策略评估误差 $\|\hat{Q}^{\pi_k} - Q^{\pi_k}\|_\infty$ 上限为 ε (见算法 6), 如果我们在 3.2 节中假设的策略改进是准确的, 则策略迭代最终生成的策略将距离最优性能很近 [Bertsekas and Tsitsiklis, 1996; Lagoudakis and Parr, 2003a] (即最优价值函数):

$$\limsup_{k \rightarrow \infty} \|Q^{\pi_k} - Q^*\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \cdot \varepsilon \quad (3.31)$$

其中 Q^* 是最优 Q 函数并且对应于最优性能, 参见第 1 章。注意, 如果执行近似策略改进, 类似的约束成立, 但是策略改进误差必须包括在不等式的右侧。

重要的一点是, 策略的顺序通常不能保证收敛到固定的策略。例如, 策略可能会以一定周期振荡。然而在公式 (3.31) 的意义上, 周期中的所有策略将具有高性能。

注意公式 (3.31) 使用无穷大范数, 而策略评估组件的边界 (公式 (3.29) 和公式 (3.30)) 使用欧几里得范数。这两种类型的边界不能容易地结合得出一个整体为近似策略迭代。[Munos, 2003] 给出了欧几里得范数的策略迭代界限, 我们在这里不详细描述。

考虑现在的策略迭代的乐观变体, 例如在线 LSPI。上述性能保证依赖于小的策略评估错误, 而在乐观情况下, 在准确的价值函数可用之前改进策略, 这意味着策略评估错误可能非常大。由于这个原因, 乐观策略迭代的动作理论上目前很难理解, 虽然算法在实践中经常工作良好。参见 [Bertsekas, 2011a; Bertsekas and Tsitsiklis, 1996] 讨论所涉及的困难。

3.6.2 有限样本的保证

上一节中报告了当样本数趋于无穷时策略评估方法的渐近性能。然而, 它们不提供关于当只有有限数量的样本可用时算法如何运行的任何保证。在本节中, 我们报告最近的有关 LSTD 和 BRM 的有限样本边界研究, 并且讨论它们在策略迭代方案中的传播机制。

虽然在前面的部分中我们关注了逼近 Q 函数的算法, 为了简单起见, 我们在这里介绍 V 函数的近似分析。符号和设置与 3.3.2 节完全相同, 我们简单地重新定义它为 V 函数。我们使用具有参数 $\theta \in \mathbb{R}^d$ 的线性近似结构和现在定义为从状态空间 S 到 \mathbb{R} 的映射的基函数 $\phi_i, i = 1, \dots, d$ 。我们用 ϕ 表示 $\phi: S \rightarrow \mathbb{R}^d, \phi(\cdot) = [\phi_1(\cdot), \dots, \phi_d(\cdot)]^\top$ BF 向量 (特征向量), 并且用 \mathcal{F} 表示 BF ϕ_i 跨越的线性函数空间, 即 $\mathcal{F} = \{f_\theta | \theta \in \mathbb{R}^d \text{ 且 } f_\theta(\cdot) = \phi^\top(\cdot)\theta\}$ 。我们将 $\tilde{\mathcal{F}}$ 定义为通过在 V_{\max} 处截断 \mathcal{F} 中的函数而获得的空间 (回想 V_{\max} 给出了最大的返回值, 上限为任何价值函数)。在所有状态下, 截断函数 \tilde{f} 等于 $f(s)$, 其中 $|f(s)| \leq V_{\max}$, 否则等于 sgn

$(f(s))V_{\max}$ 。此外, 令 L 是所有 BF 的上限, 即, 对于 $i = 1, \dots, d$, $\|\phi_i\|_{\infty} \leq L$ 。在下文中, 我们给出 LSTD-V 和 BRM-V 在有限样本上的策略评估的性能, 随后在策略评估步骤中分析使用 LSTD-V 和 BRM-V 的策略迭代算法。 V_{\max} 的截断 (truncation) 用于 LSTD-V, 但不用于 BRM-V。

1. 顺向 LSTD

令 π 为当前策略, V^{π} 为其 V 函数。令 (s_i, r_i) , $i = 1, \dots, n$ 是遵循策略 π 和 $\Phi = [\phi^{\top}(s_1); \dots; \phi^{\top}(s_n)]$; 大小为 n 的样本路径, 其中 “;” 表示矩阵中的向量 $\phi^{\top}(s_i)$ 的垂直叠加。顺向 LSTD-V 是通过如下定义经验转移矩阵 \hat{T} 而获得的 LSTD-V 的版本: 如果 $j = i+1$, $j \neq n$ 则 $\hat{T}_{ij} = 1$, 否则 $\hat{T}_{ij} = 0$ 。当应用于向量 $s = [s_1, \dots, s_n]^{\top}$, 该转换矩阵对于 $1 \leq t < n$ 且 $(\hat{T}s)_n = 0$ 返回 $(\hat{T}s)_t = s_{t+1}$ 。其余的算法和标准 LSTD 完全相同, 并返回作为线性系统 $A\theta = \gamma B\theta + b$ 解的矢量 θ , 其中 $A = \Phi^{\top}\Phi$, $B = \gamma\Phi^{\top}\hat{T}\Phi$, 并且 $b = \Phi^{\top}R$ 。类似于 3.6.1 节中使用的参数, 很容易验证经验转移矩阵 \hat{T} 导致一个收缩的经验贝尔曼算子, 因此以前的系统总是承认至少一个解决方案。虽然存在唯一的固定点, 可能存在多个解 θ 。下文中, 我们使用 $\hat{\theta}$ 来表示具有最小范数的解, 即 $\hat{\theta} = (A - \gamma B)^+ b$, 其中 $(A - \gamma B)^+$ 是矩阵 $A - \gamma B$ 的摩尔-彭罗斯伪逆矩阵^①。

[98]

对于顺向 LSTD-V 算法, 有以下性能界限 (performance bound) [Lazaric et al, 2010b]。

定理 3.1 (顺向 LSTD-V) 令 $\omega > 0$ 是格兰矩阵 $G \in \mathbb{R}^{d \times d}$ 的最小特征值, $G_{ij} = \int \phi_i^{\top}(x) \phi_j(x) \rho^{\pi}(dx)$ 。让我们假设策略 π 对具有平稳分布 ρ^{π} 的 MDP 引入一个静态的 β 混合过程 [Meyn and Tweedie, 1993]^②。存在 (s_t, r_t) , 其中 $t = 1, \dots, n$, 由 $n > n^{\pi}(\omega, \delta)$ 步通过以下策略 π 生成路径, 其中 $n^{\pi}(\omega, \delta)$ 取决于参数 ω 和 δ 的适量步长。令 $\hat{\theta}$ 为路径 LSTD-V 解, 并且 $\tilde{f}_{\hat{\theta}}$ 为其对应函数的截断, 则^③:

$$\begin{aligned} \|\tilde{f}_{\hat{\theta}} - V^{\pi}\|_{\rho^{\pi}} &\leq \frac{2}{\sqrt{1-\gamma^2}} \left(2\sqrt{2} \|V^{\pi} - \Pi^{\rho^{\pi}} V^{\pi}\|_{\rho^{\pi}} + \tilde{O} \left(L \|\theta^*\| \sqrt{\frac{d \log 1/\delta}{n}} \right) \right) \\ &\quad + \frac{1}{\sqrt{1-\gamma}} \tilde{O} \left(V_{\max} L \sqrt{\frac{d \log 1/\delta}{\omega n}} \right) \end{aligned} \quad (3.32)$$

概率为 $1-\delta$, 其中存在 θ^* 使得 $f_{\theta^*} = \Pi^{\rho^{\pi}} V^{\pi}$ 。

下面我们分析前面定理的主要条件。

- $\|V^{\pi} - \Pi^{\rho^{\pi}} V^{\pi}\|_{\rho^{\pi}}$: 这个项是近似误差, 它只取决于目标函数 V^{π} 和 \mathcal{F} 中的函数可以近似的程度。每当关于 V^{π} 的一些现有知识可使用并且 BF 被仔细设计时, 这一项可以变小。此外, 我们预计它会随着 BF 的数量 d 的减少而减少, 但是以增加边界中的其他项为代价。可以注意到, 当 n 变为无穷大时, 边界中的剩余项为

$$\frac{4\sqrt{2}}{\sqrt{1-\gamma^2}} \|V^{\pi} - \Pi^{\rho^{\pi}} V^{\pi}\|_{\rho^{\pi}}, \text{ 其匹配 LSTD 的渐近性能界限 (公式 (3.29)) 直到固定不变。}$$

① 注意, 当通用矩阵 D 可逆时, $D^+ = D^{-1}$ 。

② 大体来说, 快速混合过程是这样的, 从任意初始分布开始并遵循当前策略, 状态概率分布迅速趋向于马尔可夫链的稳定分布。

③ 为简单起见, 在 \tilde{O} 项中, 我们省略常数、 d 和 n 中的对数因子, 以及对 β 混合过程的特征参数的依赖性。

- $\tilde{O}\left(V_{\max} L \sqrt{\frac{d \log 1/\delta}{\omega n}}\right)$: 这是第一个估计误差, 这是由顺向 LSTD-V 解在有限数量的随

机采样上计算的事实引起的偏差。可以注意到, 该项会随着样本数量的降低而减少为 $n^{-1/2}$, 但是其也取决于 \mathcal{F} 中的 BF 数量 d 、BF 的 L 的范围和已有 MDP 中 V 函数的范围。该项还表现了对基于模型的格兰矩阵 G 的最小特征值 ω 的严格依赖。当 BF 是线性独立的并且矩阵 G 在稳定分布 ρ^π 下状态良好, 则 ω 必定远离 0。

- $\tilde{O}\left(L \|\theta^*\| \sqrt{\frac{d \log 1/\delta}{n}}\right)$: 这是第二估计误差, 并且类似于前一个, 它随着样本数量的

减少而减少。该估计误差不依赖于最小特征值 ω , 但是具有对 $\|\theta^*\|$ 的附加依赖性 (additional dependency), 在一些情况下可能对 \mathcal{F} 的维度有进一步的依赖性, 并且只要 BF 在静态分布 ρ^π 下不是线性无关的, 就可以取最大值。

上述的边界有助于更好地理解决定 LSTD 近似质量的主要因素 (最明显的是函数空间的一些属性)。此外, 该界限还可以用于粗略估计要达到某个期望值所需的样本数量。例如, 让我们假设近似误差为零, 并且需要误差水平 (error level) ε (即 $\|\tilde{f}_{\hat{\theta}} - V^\pi\|_{\rho^\pi}$)。在这种情况下, 建议 n 应大于 $(1-\gamma)^{-2} V_{\max}^2 L^2 d / \varepsilon^2$ (其中我们忽略在设计时不可用的项, 例如 $\|\theta^*\|$ 和 ω)。

顺向 LSTD-V 严格要求通过遵循当前策略 π (即它是策略上的) 生成样本, 有限样本分析根据稳定分布 ρ^π 来界定其性能。虽然存在无策略的 LSTD 的渐近分析 (见 3.6.1 节)。但是就我们所知, 不存在可用于无策略 LSTD 的有限样本结果。对于由 [Antos et al, 2008] 引入的改进贝尔曼残差算法, 已经获得最接近的结果, 当使用线性空间时减少到 LSTD 的无策略算法。尽管如此, [Antos et al, 2008] 提出的有限样本分析并没有从有界限扩展到线性空间, 如何界定无策略 LSTD 的性能仍然是一个悬而未决的问题。

2. 贝尔曼残差最小化

我们现在考虑寻找 V 函数的 BRM-V 算法。类似于 BRM-Q (参见 3.3.2 节), n 个样本 $(s_i, s'_{i,1}, r_{i,1}, s'_{i,2}, r_{i,2})$ 是已知的, 其中 s_i 是从任意分布 μ 中采样而来, $(s'_{i,1}, r_{i,1})$ 和 $(s'_{i,2}, r_{i,2})$ 是来自 $T(s_i, \pi(s_i), \cdot)$ 的两个独立样本, $r_{i,1}$ 、 $r_{i,2}$ 是相应的回报。该算法与 BRM-Q 类似, 但现在 BF 只是状态的函数。在介绍对 BRM-V 的性能的约束之前, 我们引入:

$$C^\pi(\mu) = (1-\gamma) \| (I - \gamma P^\pi)^{-1} \|_\mu \quad (3.33)$$

上式涉及基于 μ 和跟随策略 π 的阻尼系数状态分布的集中性系数 (例如参见 [Antos et al, 2008]), 即, 关于 μ 的 $(1-\gamma) \mu (I - \gamma P^\pi)^{-1}$ 。注意, 如果阻尼系数状态分布不是关于 μ 绝对连续, 则 $C^\pi(\mu) = \infty$ 。我们现在来介绍由 [Maillard et al, 2010] 推导出的残差范围。

定理 3.2 (BRM-V) 令 $n \geq n^\pi(\omega, \delta)$ 为的样本根据任意分布 μ 抽取独立同分布的样本数。 ω 是基于模型的格兰矩阵 $G \in \mathbb{R}^{d \times d}$ 的最小特征值, $G_{ij} = \int \phi_i^\top(x) \phi_j(x) \mu(dx)$ 。如果 $\hat{\theta}$ 是 BRM-V 的解并且 $f_{\hat{\theta}}$ 是相应的函数, 则贝尔曼残差被限制为:

$$\|f_{\hat{\theta}} - B_V^\pi f_{\hat{\theta}}\|_\mu^2 \leq \inf_{f \in \mathcal{F}} \|f - B_V^\pi f\|_\mu^2 + \tilde{O}\left(\frac{1}{\xi_\pi^2} L^4 R_{\max}^2 \sqrt{\frac{d \log 1/\delta}{n}}\right)$$

概率为 $1-\delta$, 其中 $\xi_\pi = \frac{\omega(1-\gamma)^2}{C^\pi(\mu)^2}$ 。此外, 近似误差 V^π 是:

$$\|V^\pi - f_\theta^\pi\|_\mu^2 \leq \left(\frac{C^\pi(\mu)}{1-\gamma} \right)^2 \left((1+\gamma\|P^\pi\|_\mu)^2 \inf_{f \in \mathcal{F}} \|V^\pi - f\|_\mu^2 + \tilde{O}\left(\frac{1}{\xi_\pi^2} L^4 R_{\max}^2 \sqrt{\frac{d \log 1/\delta}{n}} \right) \right)$$

我们分析定理的第一个不等式。

- $\|f_\theta^\pi - B_V^\pi f_\theta^\pi\|_\mu^2$: 不等式的左侧是 BRM-V 的解 f_θ^π 的确切贝尔曼残差。结果表明, 最小化经验贝尔曼残差是重要的, 即, 可以仅使用两个独立样本 $(s'_{i,1}, r_{i,1})$ 和 $(s'_{i,2}, r_{i,2})$ 在有限数量的状态 s_i 中计算在整个状态空间 S 中的一个小且精确的贝尔曼残差。
- $\inf_{f \in \mathcal{F}} \|f - B_V^\pi f\|_\mu^2$: 这个项是能够在空间 \mathcal{F} 中实现的最小贝尔曼残差, 并且它起到近似误差的作用。虽然难以分析, 但可以表明, 对于特定类别的 MDP, 这一项会很小, 其中回报函数和转换核都是 Lipschitz 提出的 (进一步的讨论见 [Munos and Szepesvári, 2008])。
- $\tilde{O}\left(\frac{1}{\xi_\pi^2} L^4 R_{\max}^2 \sqrt{\frac{d \log 1/\delta}{n}}\right)$: 估计误差。可以注意到, 与 LSTD 的界限不同, 这里的平方损失是有界的, 并且估计误差随着 $O(n^{-1/2})$ 速率减慢而降低。除了对 MDP 和 BF 的一些特征的依赖性, 该项还取决于与采样分布 μ 相关的格兰矩阵的最小特征值倒数的计算。虽然这种依赖关系可能很关键, 但可以通过 μ 和 BF 的仔细设计来使 ω 变大。

第二个不等式不在边界中引入其他的相关项。我们把关于 $C^\pi(\mu)$ 的讨论和对 LSTD 的更详细的比较到推迟到下一部分。

3. 策略迭代

到目前为止, 我们介绍了用于策略评估的顺向 LSTD-V 和 BRM-V 的性能界限。接下来, 我们为策略迭代算法 (LSPI 和 BRM-PI) 提供有限样本分析, 其中在每次迭代 k 中, 顺向 LSTD-V 和 BRM-V 分别用于计算 V^{π_k} 的近似值。特别地, 我们通过比较在 K 个迭代之后由这些算法返回的策略的值 V^{π_k} 和最优 V 函数 V^* 来报告性能界限。为了得到 LSPI 和 BRM-PI 的性能界限, 需要首先将 K 次迭代之后的误差 $\|V^* - V^{\pi_k}\|$ 与这些算法的每个迭代 k 后的误差相比较, 然后利用每个迭代的误差替换定理 3.1 和定理 3.2 的策略评估边界。[Antos et al, 2008] 得出了每次迭代的误差边界:

$$\|V^* - V^{\pi_k}\|_\sigma \leq \frac{4\gamma}{(1-\gamma)^2} \left(\sqrt{C_{\sigma,v}} \max_{0 \leq k < K} \|V_k - B_V^{\pi_k} V_k\|_v + \gamma^{\frac{K-1}{2}} R_{\max} \right) \quad (3.34)$$

其中 V_k 是在每次迭代时由价值函数 V^{π_k} 的算法返回的近似值, σ 和 v 是状态空间上的任意度量, 并且 $C_{\sigma,v}$ 是来自 [Antos et al, 2008] 提出的定义 2。重要的是要注意, 在这个界限中没有关于如何生成序列 V_k 的假设。在介绍顺向 LSPI 和 BRM-PI 的界限之前, 有必要做出以下假设。

假设 1 (均匀随机性) 令 μ 是状态空间上的分布。存在一个常数 C , 使得对于所有 $s \in S$, 并且对于所有 $a \in A$, $T(s, a, \cdot)$ 被 $\mu(\cdot)$ 主导, 即 $T(s, a, \cdot) \leq C\mu(\cdot)$ 。

这个假设保证不存在许多状态-动作对转移到一组有限的状态下。可以看出, 在该假设下, 公式 3.33 中定义的 $C^\pi(\mu)$ 和任何 σ 的集中性项 $C_{\sigma,\mu}$, 都是以 C 为上限的。我们现在可以得出当算法的每次迭代都使用分布 μ 且假设 1 成立时 BRM-PI 的界限。

定理 3.3 对于任何 $\delta > 0$ ，只要具有 $n \geq n(\delta/K)$ ，且 $n(\delta/K)$ 是一个恰当的样本数量，具有概率 $1-\delta$ 对于所有迭代 $1 \leq k < K$ ，存在经验贝尔曼残差最小化 f_{θ_k} 。因此，BRM-PI 算法被很好地定义，并且算法返回的策略 π_K 的性能 V^{π_K} 是这样的：

$$\|V^* - V^{\pi_K}\|_{\infty} \leq \tilde{O}\left(\frac{1}{(1-\gamma)^2} \left(C^{3/2} E_{\text{BRM}}(\mathcal{F}) + \frac{\sqrt{C}}{\xi} \left(\frac{d \log(K/\delta)}{n} \right)^{1/4} + \gamma^{K/2} \right)\right)$$

其中 $E_{\text{BRM}}(\mathcal{F}) = \sup_{\pi \in \mathcal{F}(\mathcal{F})} \inf_{f \in \mathcal{F}} \|f - V^{\pi}\|_{\mu}$ ， $\mathcal{F}(\mathcal{F})$ 是有关 \mathcal{F} 中函数的所有贪婪策略的集合，并且 $\xi = \frac{\omega(1-\gamma)^2}{C^2} \leq \xi_{\pi}$ ， $\pi \in \mathcal{F}(\mathcal{F})$ 。

在这个定理中， $E_{\text{BRM}}(\mathcal{F})$ 查看策略 π 的 V 函数的最小近似误差，并且从 \mathcal{F} 的一些近似 V 函数中的贪婪策略集合中取该误差的最坏情况。第二项 $\frac{\sqrt{C}}{\xi} \left(\frac{d \log(K/\xi)}{n} \right)^{1/4}$ 是估计误差，它

102 包含与定理 3.2 中相同的项，它减少为 $\tilde{O}(d/n)$ 。最后，最后一项 $\gamma^{K/2}$ 简单地考虑了由于有限迭代次数而导致的误差，随着 K 增加而迅速变为零。

现在我们转向顺向 -LSPI 并给出该算法的性能界限。在顺向 LSPI 的每个迭代 k 中，通过遵循由所评估的策略 π_k 生成的单个轨迹来收集样本，并且使用顺向 LSTD 来计算 V^{π_k} 的近似。把顺向 LSTD 界限（定理 3.1）插入到公式 (3.34) 中，应该首先注意到 $\|V_k - B_V^{\pi_k} V_k\|_{\rho^{\pi_k}} \leq (1+\gamma) \|V_k - V^{\pi_k}\|_{\rho^{\pi_k}}$ 。使用这种方式代替贝尔曼残差，我们在每次迭代时使用近似误差 $V_k - V^{\pi_k}$ 来划定算法的性能界限。还要注意的，顺向 LSTD 界限需要遵循所评估的策略来收集样本。这可能在迭代序列中引入严重的问题。事实上，如果一个策略过多地集中在小部分的状态空间上，即使策略评估在那些状态上是准确的，对于当前策略未覆盖过的状态的性能可能是任意坏的。结果，策略改进可能产生不良策略，这可能反过来导致任意不良策略迭代过程。因此，这里需要进行以下假设。

假设 2 (下限分布) 令 μ 是状态空间上的分布。对任何策略 π ，截断空间 $\tilde{\mathcal{F}}$ 是贪婪的， $\mu(\cdot) \leq \kappa \rho^{\pi}(\cdot)$ ，其中 $k < \infty$ 是常数， ρ^{π} 是策略 π 的稳定分布。

还需要以高概率保证在顺向 LSPI 算法的每次迭代中存在唯一的顺向 LSTD 解，因此，需要以下假设。

假设 3 (线性无关 BF) 令 μ 是假设 2 的下界分布。我们假设函数空间 \mathcal{F} 的 $BF \phi(\cdot)$ 是关于 μ 线性独立的。在这种情况下，格兰矩阵 $G_{\mu} \in \mathbb{R}^{d \times d}$ 的最小特征值 ω_{μ} 严格为正。

假设 2 和假设 3 (加上在执行算法期间观察到的 β 混合过程的特征参数的一些小的假设) 足以给出顺向 LSPI 算法的性能界限。然而，为了更容易与 BRM-PI 的范围进行比较，我们还假设“假设 1”适用于顺向 LSPI。

103 **定理 3.4** 假设在顺向 LSPI 算法的每次迭代 k 中，从具有固定分布 $\rho_{k-1} = \rho^{\pi_{k-1}}$ 的固定 β 混合过程生成大小为 $n > n(\omega_{\mu}, \delta)$ 的路径。设 $V_{-1} \in \tilde{\mathcal{F}}$ 为任意初始 V 函数， V_0, \dots, V_{K-1} ($\tilde{V}_0, \dots, \tilde{V}_{K-1}$) 在 K 个迭代之后由顺向 LSPI 生成，并且 π_K 是截断的 V 函数 \tilde{V}_{K-1} 的贪婪策略。然后在假设 1 ~ 3 和在执行算法期间观察到的 β 混合过程的特征参数的一些假设下，具有概率 $1-\delta$ ，我们有：

$$\|V^* - V^{\pi_K}\|_{\infty} \leq \tilde{O}\left(\frac{1}{(1-\gamma)^2} \left(\frac{\sqrt{CK}}{1-\gamma} E_{\text{LSTD}}(\mathcal{F}) + \frac{\kappa \sqrt{C}}{\sqrt{\omega_{\mu}}} \left(\frac{d \log(K/\delta)}{n} \right)^{1/2} + \gamma^{K/2} \right)\right)$$



其中 $E_{\text{LSTD}}(\mathcal{F}) = \sup_{\pi \in \mathcal{G}(\mathcal{F})} \inf_{f \in \mathcal{F}} \|f - V^\pi\|_{\rho^\pi}$, 是关于所有 \mathcal{F} 函数的贪婪策略。

请注意, 初始策略 π_0 在 V 函数 V_{-1} 中是贪婪的, 而不是随意的, 这就是为什么我们在初始 V 函数中使用索引 -1 的原因。

比较 BRM-PI 和顺向 LSPI 的性能界限, 我们首先注意到 BRM-PI 具有的估计速率是 $O(n^{-1/4})$, 比 $O(n^{-1/2})$ 还差。我们还可以看到, BRM-PI ($E_{\text{BRM}}(\mathcal{F})$) 中的近似误差项不如顺向 LSPI ($E_{\text{LSTD}}(\mathcal{F})$) 的复杂, 因为 $E_{\text{BRM}}(\mathcal{F})$ 中的范数仅与分布 μ 相关, 而 $E_{\text{LSTD}}(\mathcal{F})$ 中的一个与任何策略在 $\mathcal{G}(\mathcal{F})$ 中的平稳分布相关。算法使用的假设也不同。在 BRM-PI 中, 假定生成模型是可用的, 因此, 可以在任何采样分布 μ 下获得性能界限, 具体如假设 1 所说。另一方面, 在顺向 LSPI 的每次迭代时, 需要使用根据所评估的策略生成的单个轨迹。这可以在该策略的稳定分布下提供性能界限, 并且在策略迭代方案中准确地估计当前策略可能是不够的, 因为关于该近似的贪婪策略可能是任意差的 (arbitrarily poor)。因此, 我们可以得出结论, BRM-PI 的性能比顺向 LSPI 的性能更易控制。这反映在如下事实中, 即在 BRM-PI 中可以通过仅选择统一的主导分布 μ (假设 1), 例如均匀分布; 而在顺向 LSPI 中, 我们需要对在算法的迭代中遇到的策略的稳定分布做出更强的假设, 例如满足 (假设 2) 下限的均匀分布。

3.7 延伸阅读

研究人员已经提出了在 3.3 节中介绍的方法的许多扩展和变体, 但遗憾的是, 我们没有足够的篇幅来描述它们。在本节中, 我们将 (非详尽地) 提及这些研究的亮点, 为更多对细节感兴趣的读者提供文献帮助。

如前所述, 可以使用多阶段贝尔曼映射做近似策略估计。该映射由 $\lambda \in [0, 1)$ 参数化, Q 函数由下式给出:

104

$$B_{Q,\lambda}^\pi(Q) = (1-\lambda) \sum_{t=0}^{\infty} \lambda^t (B_Q^\pi)^{t+1}(Q)$$

其中 $(B_Q^\pi)^t$ 表示 B_Q^π 与其自身的 t 次组成。在本章中, 我们只考虑了单次迭代的情况, 其中 $\lambda = 0$, 但事实上, 通常在 $\lambda \in [0, 1)$ 的情况下讨论近似策略评估, [Nedić and Bertsekas, 2003; Yu, 2010; Bertsekas and Ioffe, 1996; Yu and Bertsekas, 2009] 以及第 1 章中 $\text{TD}(\lambda)$ 算法的讨论。注意, 在原始 LSPI 中, 非零 λ 将阻止样本重用; 相反, 在每次迭代时, 必须使用当前策略生成新的样本。

非参数逼近器不是预定义的, 而是从数据自动构建的, 因此在很大程度上它们使用户免于设计 BF 的困扰。非参数技术主要是基于核的近似, 例如, [Xu et al, 2005, 2007; Jung and Polani, 2007b; Farahmand et al, 2009] 提出其与 LSTD 的组合, [Jung and Polani, 2007a, b] 提出其与 LSPE 的组合, [Farahmand et al, 2009] 提出其与 BRM 的组合。高斯过程的相关框架也已用于策略评估 [Rasmussen and Kuss, 2004; Engel et al, 2005; Taylor and Parr, 2009]。在它们的基本形式中, 基于核的方法和高斯过程的计算需求随着所考虑的样本的数量增长而增长。由于这个数在实践中可能很大, 上述许多方法使用内核稀疏化技术来限制对解决方案有贡献的样本数量 [Xu et al, 2007; Engel et al, 2003, 2005; Jung and Polani, 2007a, b]。

与稀疏化密切相关的是正则化技术, 其控制解决方案的复杂性 [Farahmand et al, 2009; Kolter and Ng, 2009]。例如, 为了获得 LSTD 的正则化变形, 可添加一个惩罚项到投影策略



评估问题(公式(3.8))中,得到:

$$\min_{\hat{Q} \in \mathcal{Q}} \left[\left\| \hat{Q} - \Pi(B_Q^\pi(\hat{Q})) \right\| + \beta v(\hat{Q}) \right]$$

其中 β 是正的正则化系数, v 是随着 \hat{Q} 的复杂度而增长的惩罚函数。(例如,逼近器的参数向量的范数)。注意, [Farahmand et al, 2009] 还对投影 Π 添加了类似的正则化项, 并且另外讨论了正则化版本的 BRM。

为了减少 LSTD 的计算成本, 有人提出了所谓的增量变体, 其在给定的迭代中仅更新少数参数 [Geramifard et al, 2006, 2007]。

LSPE 的一般化的缩放变体, 由 [Bertsekas, 2011a] 提出, 可以通过首先将 LSPE 更新(公式(3.20))重写为等价形式(假设 A 是可逆的)来获得:

105

$$\theta_{t+1} = \theta_t - \alpha A^{-1}[(A - \gamma B)\theta_t - b]$$

然后用一个更一般化的缩放矩阵 Γ 替换 A^{-1} :

$$\theta_{t+1} = \theta_t - \alpha \Gamma[(A - \gamma B)\theta_t - b]$$

在选择适当的 Γ 和 α 下, 该算法收敛的条件比原来的 LSPE 条件更宽松。

在 BRM 的背景下, [Antos et al, 2008] 通过修改 BRM 解决的最小化问题消除了对双采样的需求。他们表明, 这个修改过的问题中的单一样本估计的系数是无偏的, 而解决方案始终有意义。

我们还注意到一些策略评估和迭代的最小二乘法的替代方案, 特别是基于梯度更新的技术 [Sutton et al, 2009b, a; Maei et al, 2010] 和蒙特卡罗模拟 [Lagoudakis and Parr, 2003b; Dimitrakakis and Lagoudakis, 2008; Lazaric et al, 2010a]。

虽然 LSTD、LSPE 和 BRM 策略评估的统一视角和理论研究迄今为止还没有在文献中使用, 但是强化学习和动态规划方法在最近的各种文献中已经被提及。例如, 感兴趣的读者应该知道 [Buşoniu et al, 2010a] 的第 3 章描述了 LSTD-Q 和 LSPE-Q 作为近似强化学习和动态规划的介绍的一部分, [Munos, 2010] 涉及 LSTD 和 BRM, [Szepesvári, 2010] 的第 3 章概述了 LSTD 和 LSPE 方法, [Bertsekas, 2011a, 2010] 在很大程度上涉及这两种类型的方法。

本书的第 7 章进一步介绍了近似强化学习领域的更一般性的观点, 而非关注最小二乘法。

参考文献

- Antos, A., Szepesvári, C., Munos, R.: Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning* 71(1), 89–129 (2008)
- Baird, L.: Residual algorithms: Reinforcement learning with function approximation. In: *Proceedings 12th International Conference on Machine Learning (ICML-1995)*, Tahoe City, U.S., pp. 30–37 (1995)
- Bertsekas, D.P.: A counterexample to temporal differences learning. *Neural Computation* 7, 270–279 (1995)
- Bertsekas, D.P.: Approximate dynamic programming. In: *Dynamic Programming and Optimal Control*, Ch. 6, vol. 2 (2010), <http://web.mit.edu/dimitrib/www/dpchapter.html>
- Bertsekas, D.P.: Approximate policy iteration: A survey and some new methods. *Journal of Control Theory and Applications* 9(3), 310–335 (2011a)
- Bertsekas, D.P.: Temporal difference methods for general projected equations. *IEEE Transactions on Automatic Control* 56(9), 2128–2139 (2011b)



- Bertsekas, D.P., Ioffe, S.: Temporal differences-based policy iteration and applications in neuro-dynamic programming. Tech. Rep. LIDS-P-2349, Massachusetts Institute of Technology, Cambridge, US (1996), <http://web.mit.edu/dimitrib/www/Tempdif.pdf>
- Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific (1996)
- Bertsekas, D.P., Borkar, V., Nedić, A.: Improved temporal difference methods with linear function approximation. In: Si, J., Barto, A., Powell, W. (eds.) Learning and Approximate Dynamic Programming. IEEE Press (2004)
- Boyan, J.: Technical update: Least-squares temporal difference learning. Machine Learning 49, 233–246 (2002)
- Bradtke, S.J., Barto, A.G.: Linear least-squares algorithms for temporal difference learning. Machine Learning 22(1-3), 33–57 (1996)
- Buşoniu, L., Babuška, R., De Schutter, B., Ernst, D.: Reinforcement Learning and Dynamic Programming Using Function Approximators. In: Automation and Control Engineering. Taylor & Francis, CRC Press (2010a)
- Buşoniu, L., De Schutter, B., Babuška, R., Ernst, D.: Using prior knowledge to accelerate online least-squares policy iteration. In: 2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR-2010), Cluj-Napoca, Romania (2010b)
- Buşoniu, L., Ernst, D., De Schutter, B., Babuška, R.: Approximate dynamic programming with a fuzzy parameterization. Automatica 46(5), 804–814 (2010c)
- Buşoniu, L., Ernst, D., De Schutter, B., Babuška, R.: Online least-squares policy iteration for reinforcement learning control. In: Proceedings 2010 American Control Conference (ACC-2010), Baltimore, US, pp. 486–491 (2010d)
- Dimitrakakis, C., Lagoudakis, M.: Rollout sampling approximate policy iteration. Machine Learning 72(3), 157–171 (2008)
- Engel, Y., Mannor, S., Meir, R.: Bayes meets Bellman: The Gaussian process approach to temporal difference learning. In: Proceedings 20th International Conference on Machine Learning (ICML-2003), Washington, US, pp. 154–161 (2003)
- Engel, Y., Mannor, S., Meir, R.: Reinforcement learning with Gaussian processes. In: Proceedings 22nd International Conference on Machine Learning (ICML-2005), Bonn, Germany, pp. 201–208 (2005)
- Ernst, D., Geurts, P., Wehenkel, L.: Tree-based batch mode reinforcement learning. Journal of Machine Learning Research 6, 503–556 (2005)
- Farahmand, A.M., Ghavamzadeh, M., Szepesvári, C.S., Mannor, S.: Regularized policy iteration. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, vol. 21, pp. 441–448. MIT Press (2009)
- Geramifard, A., Bowling, M.H., Sutton, R.S.: Incremental least-squares temporal difference learning. In: Proceedings 21st National Conference on Artificial Intelligence and 18th Innovative Applications of Artificial Intelligence Conference (AAAI-2006), Boston, US, pp. 356–361 (2006)
- Geramifard, A., Bowling, M., Zinkevich, M., Sutton, R.S.: iLSTD: Eligibility traces & convergence analysis. In: Schölkopf, B., Platt, J., Hofmann, T. (eds.) Advances in Neural Information Processing Systems, vol. 19, pp. 440–448. MIT Press (2007)
- Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. Johns Hopkins (1996)
- Jung, T., Polani, D.: Kernelizing LSPE(λ). In: Proceedings 2007 IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL-2007), Honolulu, US, pp. 338–345 (2007a)
- Jung, T., Polani, D.: Learning RoboCup-keepaway with kernels. In: Gaussian Processes in Practice, JMLR Workshop and Conference Proceedings, vol. 1, pp. 33–57 (2007b)
- Kolter, J.Z., Ng, A.: Regularization and feature selection in least-squares temporal difference learning. In: Proceedings 26th International Conference on Machine Learning (ICML-2009), Montreal, Canada, pp. 521–528 (2009)
- Konda, V.: Actor-critic algorithms. PhD thesis, Massachusetts Institute of Technology, Cambridge, US (2002)
- Lagoudakis, M., Parr, R., Littman, M.: Least-squares Methods in Reinforcement Learning for Control. In: Vlahavas, I.P., Spyropoulos, C.D. (eds.) SETN 2002. LNCS (LNAI),



- vol. 2308, pp. 249–260. Springer, Heidelberg (2002)
- Lagoudakis, M.G., Parr, R.: Least-squares policy iteration. *Journal of Machine Learning Research* 4, 1107–1149 (2003a)
- Lagoudakis, M.G., Parr, R.: Reinforcement learning as classification: Leveraging modern classifiers. In: *Proceedings 20th International Conference on Machine Learning (ICML-2003)*, Washington, US, pp. 424–431 (2003b)
- Lazaric, A., Ghavamzadeh, M., Munos, R.: Analysis of a classification-based policy iteration algorithm. In: *Proceedings 27th International Conference on Machine Learning (ICML-2010)*, Haifa, Israel, pp. 607–614 (2010a)
- Lazaric, A., Ghavamzadeh, M., Munos, R.: Finite-sample analysis of LSTD. In: *Proceedings 27th International Conference on Machine Learning (ICML-2010)*, Haifa, Israel, pp. 615–622 (2010b)
- Li, L., Littman, M.L., Mansley, C.R.: Online exploration in least-squares policy iteration. In: *Proceedings 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2009)*, Budapest, Hungary, vol. 2, pp. 733–739 (2009)
- Maei, H.R., Szepesvári, C., Bhatnagar, S., Sutton, R.S.: Toward off-policy learning control with function approximation. In: *Proceedings 27th International Conference on Machine Learning (ICML-2010)*, Haifa, Israel, pp. 719–726 (2010)
- Maillard, O.A., Munos, R., Lazaric, A., Ghavamzadeh, M.: Finite-sample analysis of Bellman residual minimization, vol. 13, pp. 299–314 (2010)
- Meyn, S., Tweedie, L.: *Markov chains and stochastic stability*. Springer, Heidelberg (1993)
- Moore, A.W., Atkeson, C.R.: The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning* 21(3), 199–233 (1995)
- Munos, R.: Error bounds for approximate policy iteration. In: *Proceedings 20th International Conference (ICML-2003)*, Washington, US, pp. 560–567 (2003)
- Munos, R.: *Approximate dynamic programming*. In: *Markov Decision Processes in Artificial Intelligence*. Wiley (2010)
- Munos, R., Szepesvári, C.S.: Finite time bounds for fitted value iteration. *Journal of Machine Learning Research* 9, 815–857 (2008)
- Nedić, A., Bertsekas, D.P.: Least-squares policy evaluation algorithms with linear function approximation. *Discrete Event Dynamic Systems: Theory and Applications* 13(1-2), 79–110 (2003)
- Rasmussen, C.E., Kuss, M.: Gaussian processes in reinforcement learning. In: Thrun, S., Saul, L.K., Schölkopf, B. (eds.) *Advances in Neural Information Processing Systems*, vol. 16. MIT Press (2004)
- Scherrer, B.: Should one compute the Temporal Difference fix point or minimize the Bellman Residual? the unified oblique projection view. In: *Proceedings 27th International Conference on Machine Learning (ICML-2010)*, Haifa, Israel, pp. 959–966 (2010)
- Schweitzer, P.J., Seidmann, A.: Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications* 110(2), 568–582 (1985)
- Sutton, R., Maei, H., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C.S., Wiewiora, E.: Fast gradient-descent methods for temporal-difference learning with linear function approximation. In: *Proceedings 26th International Conference on Machine Learning (ICML-2009)*, Montreal, Canada, pp. 993–1000 (2009a)
- Sutton, R.S.: Learning to predict by the method of temporal differences. *Machine Learning* 3, 9–44 (1988)
- Sutton, R.S., Szepesvári, C.S., Maei, H.R.: A convergent $O(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) *Advances in Neural Information Processing Systems*, vol. 21, pp. 1609–1616. MIT Press (2009b)
- Szepesvári, C.S.: *Algorithms for Reinforcement Learning*. Morgan & Claypool Publishers (2010)
- Taylor, G., Parr, R.: Kernelized value function approximation for reinforcement learning. In: *Proceedings 26th International Conference on Machine Learning (ICML-2009)*, Mon-



- treal, Canada, pp. 1017–1024 (2009)
- Thiery, C., Scherrer, B.: Least-squares λ policy iteration: Bias-variance trade-off in control problems. In: Proceedings 27th International Conference on Machine Learning (ICML-2010), Haifa, Israel, pp. 1071–1078 (2010)
- Tsitsiklis, J.N.: On the convergence of optimistic policy iteration. *Journal of Machine Learning Research* 3, 59–72 (2002)
- Tsitsiklis, J.N., Van Roy, B.: An analysis of temporal difference learning with function approximation. *IEEE Transactions on Automatic Control* 42(5), 674–690 (1997)
- Xu, X., Xie, T., Hu, D., Lu, X.: Kernel least-squares temporal difference learning. *International Journal of Information Technology* 11(9), 54–63 (2005)
- Xu, X., Hu, D., Lu, X.: Kernel-based least-squares policy iteration for reinforcement learning. *IEEE Transactions on Neural Networks* 18(4), 973–992 (2007)
- Yu, H.: Convergence of least squares temporal difference methods under general conditions. In: Proceedings 27th International Conference on Machine Learning (ICML-2010), Haifa, Israel, pp. 1207–1214 (2010)
- Yu, H., Bertsekas, D.P.: Convergence results for some temporal difference methods based on least squares. *IEEE Transactions on Automatic Control* 54(7), 1515–1531 (2009)
- Yu, H., Bertsekas, D.P.: Error bounds for approximations from projected linear equations. *Mathematics of Operations Research* 35(2), 306–329 (2010)

107
110



第4章

Reinforcement Learning: State-of-the-Art

学习和使用模型

Todd Hester, Peter Stone

摘要

相对于无模型的、其直接从域中的经验进行学习的强化学习方法，基于模型的强化学习方法在线学习一个转换函数和一个回报函数，并且使用这个模型去设计一个策略。一旦这个方法学习到了一个精确的模型，它可以在不需要任何额外经验知识的情况下设计一个最优策略。因此，当基于模型的方法能够快速的学习一个好的模型，相比于无模型方法，它们常常提升了样本效率，而无模型方法需要采取动作去把值反馈到先前的状态。基于模型的方法的另外一个优点是，它们可以使用它们的模型来规划多步的探索轨迹。特别的是，许多方法驱动学习器去探索模型中不确定的地方，以便更快速地学习模型。在本章中，我们回顾一些使用基于模型的方法，以及在这些模型上进行规划的方法。此外，我们研究联合模型学习和规划的典型结构，这取决于设计者想要使用算法去线上运行、批处理模型执行或者实时运行。这些算法的主要性能标准之一就是样本的复杂性，或该算法必须采取多少动作去完成学习。我们探究一些算法的样本效率，这高度依赖于算法是否拥有智能的探索机制。我们回顾了一些方法来解决探索问题，包括贝叶斯方法，它保持着一个在可能模型上的信念分布去明确地测量模型中的不确定性。我们在两个实例域上展示了基于模型的方法和无模型方法的一些经验比较，得出结论。最后，总结并回顾了一些可以扩展到更大的域上改进样本和计算的方法。

111

4.1 简介

到目前为止，前面书中提及的都是无模型的强化学习方法，该算法直接通过域的经验来更新价值函数。然而，基于模型的方法（或间接方法）是从域的模型中执行它的更新，而不是从域本身。模型从域中的经验进行学习，然后通过在学习到的模型价值函数上进行规划。该过程如图 4.1 所示。这种规划可以在模型上简单地运行一个无模型的方法，或者它可以是诸如值迭代或者蒙特卡罗树搜索的方法。

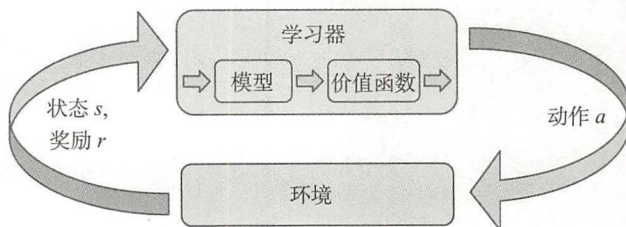


图 4.1 基于模型的强化学习学习器使用它们的经验首先学习域的模型，然后用这个模型来计算价值函数



由这些方法学习到的模型可以有很大的不同。模型可以完全从头开始学习，可以给出模型的结构以便只有参数需要学习，或者提供一个近乎完整的模型。如果算法能够足够快地学习到一个正确的模型，那么比起无模型的方法，基于模型的强化学习能够使得样本更高效（采取更少的动作去学习）。一旦学习到一个准确的模型，无需任何额外的经验，可以得到一个最优策略。比如，当一个学习器首先发现一个目标状态，它的策略值可以通过规划该目标的新模型来立即更新。相反的，一个无模型的方法必须多次跟踪目标的轨迹，以使值一直传播返回到开始状态。这种更高的样本效率通常带来更多的模型学习和规划策略的计算开销以及更多的空间来表示该模型。

模型的另外一个优点是它们为学习器进行目标探索提供了机会。学习器可以通过模型来规划一个策略，以驱动学习器探索特定的状态，这些状态可以是没有被访问的或者不确定的状态。快速学习模型的一个关键是获得正确的经验去学习模型（类似于主动学习）。人们已经提出了多种探索的方法，引导了准确模型的快速学习，从而具有更高的样本效率。 [112]

基于模型的强化学习的主要组成部分是模型，在 4.2 节中介绍，同时，在 4.3 节中描述了规划模型策略的规划器。4.4 节讨论了如何将模型和规划器组合成一个完整的学习器。算法的样本复杂性（在 4.5 节中解释）是这些方法进行评估的主要性能标准之一。在 4.6 节中，我们研究因子域的算法。探索是提高样本复杂性的主要焦点之一，在 4.7 节中讲解。在 4.8 节中，我们讨论对连续域的扩展，在 4.9 节中经验性地测试了一些算法的性能，并在 4.10 节中把讨论扩展到更大和更现实的问题中。最后，在 4.11 节进行了总结。

4.2 什么是模型

模型是学习器所需要的信息，旨在模拟在马尔可夫决策过程（MDP）中采取一个动作的结果。如果学习器从状态 s 采取动作 a ，模型就需要预测下一个状态 s' 和回报 r 。如果域是随机的，模型可以提供在下一个状态上的完整概率分布，或者它可能是一个生成模型，简单地提供下一个状态分布的样本。当学习器与其环境交互时，该模型被更新，然后与规划算法一起使用去计算一个策略。

一种常见的模型学习方法是去学习一个最大似然模型列表。在这种情况下，算法保持一个计数 $C(s,a)$ ，表示从状态 s 执行动作 a 所消耗的次数， $C(s,a,s')$ 表示从 (s,a) 到每一个状态 s' 的次数。 s' 的概率是：

$$P(s'|s,a) = T(s,a,s') = C(s,a,s')/C(s,a)$$

算法也保存每一次转换的回报和 $Rsum(s,a)$ ，将特定状态-动作的预期回报计算为从该状态-动作接收的平均回报：

$$R(s,a) = Rsum(s,a)/C(s,a)$$

这个表格模型学习是十分简单的，并且在一些常见的基于模型的算法中使用，比如 R-MAX [Brafman and Tennenholtz, 2011]。

表 4.1 展示了如图 4.2 所示的域中经过 50 次之后的表格模型的计数和预测的示例。基于每一次经验转换的计数，模型可以对实际的转换概率 $T(s,a,s')$ 和回报 $R(s,a)$ 进行预测。模型的预测对来自状态 1 的动作 0 是十分准确的，而其他的状态-动作对之间的概率是近似正确的，但随着更多的样本而改善。

学习域的表格模型可以看作在域中学习每个状态-动作对的一个单独模型，它需要学习器采取许多动作来学习模型。如果我们假设转换动力学类似于状态-动作对，我们可以通过



在模型学习中合并泛化来改进表格模型。模型学习可以看作是一个监督学习的问题， (s,a) 是输入， s' 和 r 是监督学习者试图预测的输出。合并泛化的好处之一是，监督学习者基于已经训练过的转换对看不见的状态的模型进行预测。

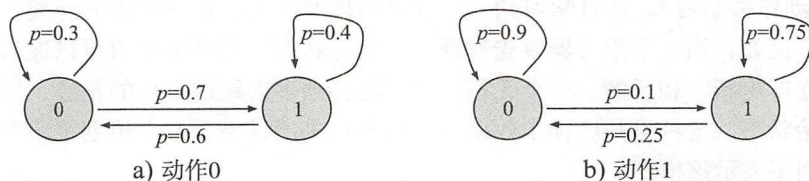


图 4.2 该图展示了一个简单的两状态 MDP 动作 0 和动作 1 的转换概率。我们假设一个 -1 回报的转换导致状态 0，+1 回报的转换导致状态 1

表 4.1 展示了如图 4.2 所示的域中的 50 次经验之后所学的模型的示例

状态	动作	$C(s,a)$	$C(s,a,0)$	$C(s,a,1)$	$RSum(s,a)$	$T(s,a,0)$	$T(s,a,1)$	$R(s,a)$
0	0	12	4	8	4	0.33	0.67	0.33
0	1	14	13	1	-12	0.93	0.07	-0.86
1	0	10	6	4	-2	0.6	0.4	-0.2
1	1	14	4	10	6	0.29	0.71	0.43

自适应动态规划领域的许多方法都采用神经网络去学习域的模型 [Prokhorov and Wunsch, 1997; Venayagamoorthy et al, 2002]。这些方法训练神经网络时，以前一个状态值来估计下一个状态值。神经网络模型在一个行演员 - 评论家框架内使用，模型的预测值用作估计给定状态的值的评论家神经网络的输入。

采用监督学习技术进行学习的模型基于已经被训练的状态 - 动作对，对来自未知的状态 - 动作对的转换做出预测。在许多域中，很容易地概括出转换的相对效应而不是绝对的结果。这相对的转换作用 s' 是指下一个状态到当前状态的差异：

$$s' = s' - s$$

比如，在机器人控制任务中，很容易地概括出给定的动作增大关节的角度，而不是试图概括绝对动作后关节角度的值。这种方法在许多算法中都进行了使用。随机森林被用来模拟相对转换效应 [Hester and Stone, 2010]。基于最近邻居的相对效应的均值，[Jong and Stone, 2007] 对一个给出的状态 - 动作对做出了预测。

最早的基于模型的强化学习方法之一使用局部加权回归 (LWR) 去学习模型 [Schaal and Atkeson, 1994; Atkeson et al, 1997]。学习器的所有经验都保存在内存中，当查询一个给定的状态 - 动作时，局部加权回归模型会提供一个被查询的状态 - 动作对的平均下一个状态的预测。结合独特的探索机制，这个算法能够学会控制机器人杂耍。

4.3 规划

一旦学习器学习了一个域动态的近似模型，该模型可以用来学习一个改进的策略。通常地，学习器将在模型每次更改时重新规划模型中的策略。基于模型计算策略就叫作规划。这些方法可用作在一个提供的模型上规划一个策略，而不是在在线学习模型时进行规划。在模型上进行规划的一个选项是使用第 1 章描述的动态规划方法，比如值迭代或者策略迭代。另一个方式是使用蒙特卡罗方法，特别是下面描述的蒙特卡罗搜索树 (MCTS) 方法。两类方

法的主要不同是动态规划方法计算整个状态空间的价值函数，然而 MCTS 计算集中在学习器很可能遇到的状态。

蒙特卡罗方法

第1章描述了如何在与环境交互时使用蒙特卡罗方法来计算策略。这些方法可以以类似的方式用于使用学习模型模拟的经验。该方法模拟经验的完整轨迹，直到一个片段的结束或者达到最大搜索深度，每一个模拟的轨迹叫作一个滚动。之后，它们沿着该轨迹将状态的值更新为在访问该状态之后接收到的折扣的总和。这些方法只需要环境的生成模型，而不是下一个状态的完全分布。现已存在各种各样的 MCTS 方法，不同点在于它们在搜索中在每个状态时选择动作的方式。

如图 4.3 所示，蒙特卡罗搜索树方法建立了一棵从当前状态到访问的状态-动作对的树。该树使得算法重用以前访问过的状态的信息。在朴素 MCTS 中，算法对指定树中的深度采用贪婪动作，然后从那儿采取随机动作，直到片段结束。树的搜索将值更新集中在 MCTS 在学习器的当前状态和滚动的结束之间访问的状态，这个比动态规划方法在整个状态空间上规划要更加高效。我们将讨论 MCTS 的几个变体，主要不同在于它们在每个状态如何选择动作去寻找一个好的策略。

[115]

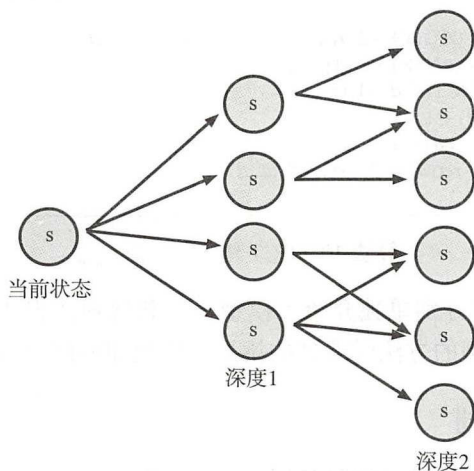


图 4.3 展示了从学习器当前状态到深度为 2 的蒙特卡罗树搜索的延伸。MCTS 方法模拟从学习器的当前状态向前的轨迹。在每个状态，它们都选择某个活动，使得它们到达树的更深一层的下一个状态。在推出一个终端状态或最大深度之后，选中的活动的值沿着上述轨迹接收回报的方向更新

稀疏采样 [Kearns et al, 1999] 是本章节讨论的 MCTS 规划方法的先驱。作者基于域中的最大回报 R_{\max} 和阻尼系数 γ 来确定准确地估计给定的状态-动作值 (s, a) 的下一个状态的样本数量 C 。他们还确定在给定阻尼系数 γ 的情况下必须搜索的水平线 h 。基于在深度 $t+1$ 处的下一个状态的 C 个样本，估计在深度 t 的一个状态-动作对的值。每一个下一个状态的值是基于在那个状态每个动作的 C 个样本，之后达到 h 。不同于 MCTS 所做的采用轨迹，这种方法在每一步扩展至更深一层直达到计算边界。该算法的运行时间为 $O((|A|C)^h)$ 。作为一种被证明能收敛到准确值的基于采样的规划方法，稀疏采样为随后的 MCTS 方法提供了重要的理论基础。

[116]

UCT [Kocsis and Szepesvari, 2006] 是 MCTS 算法的另一个变体, 通过将其样本集中在最有希望的动作上来减少稀疏采样的运行时间。UCT 从开始状态搜索到结束, 使用 UCB1 算法 [Auer et al, 2002] 基于上置信界限选择动作。该算法维持一个计数 $C(s, d)$, 即在搜索处 d 的一个给定的搜索深度访问每个状态的次数, 同时维持一个计数 $C(s, a, d)$, 即从该深度状态开始采取的动作 a 的次数。这些计数用于计算上置信界限去选择动作。在每个步骤动作选择由以下公式计算 (其中, C_p 是一个在域中回报范围内合适的常量):

$$a = \operatorname{argmax}_a Q^d(s, a) + 2C_p \sqrt{\frac{\log(C(s, d))}{C(s, a, d)}}$$

通过使用置信区间的上限来选择动作, 该算法主要采样良好的动作, 同时仍然探索何时其他动作有更高的置信界限。算法 10 展示了 UCT 算法的伪代码, 它从学习器的当前状态 s 、深度 d 为 0 开始运行。该算法提供了一个学习率 α 和域中每一步回报的范围 r_{range} 。算法的第 6 行递归调用 UCT 方法, 以便在搜索树中的下一个状态更深一层去采样一个动作。UCT 的修改版本在 Go 算法作为游戏模型的规划器中取得巨大成功 [Wang and Gelly, 2007], 也在基于模型的强化学习算法中作为规划器使用 [Sliver et al, 2008; Hester and Stone, 2010]。

```

1: if TERMINAL or d == MAXDEPTH then
2:   return 0
3:  $a \leftarrow \operatorname{argmax}_{a'} (Q^d(s, a') + 2 \cdot r_{\text{range}} \cdot \sqrt{\log(c(s, d)) / c(s, a', d)})$ 
4:  $(s', r) \leftarrow \text{SAMPLENEXTSTATE}(s, a)$ 
5:  $\text{retval} \leftarrow r + \text{UCT}(s', d + 1, \alpha)$ 
6:  $c(s, d) \leftarrow c(s, d) + 1$ 
7:  $c(s, a, d) \leftarrow c(s, a, d) + 1$ 
8:  $Q^d(s, a') \leftarrow \alpha \cdot \text{retval} + (1 - \alpha) \cdot Q(s, a', d)$ 
9: return retval

```

算法 10 UCT 输入 $s, d, a, r_{\text{range}}$

我们在 4.2 节和 4.3 节分别单独介绍了模型学习和规划的想法, 现在, 我们讨论将两者结合为一个完整的基于模型的方法时遇到的挑战, 以及如何解决这些挑战。

4.4 联合模型和规划

这里有许多方式将模型学习和规划进行结合。通常, 当学习器和环境交互时, 其模型都在每个时间步骤用最新的转换 $\langle s, a, r, s' \rangle$ 来进行更新。每次模型更新时, 算法都会使用规划器重新规划 (如图 4.4 所示)。这个方法已经被许多算法使用 [Brafman and Tennenholtz, 2001; Hester and Stone, 2009; Degris et al, 2006]。然而, 由于学习模型和在其上进行规划的计算复杂性, 它并不总是可行的。

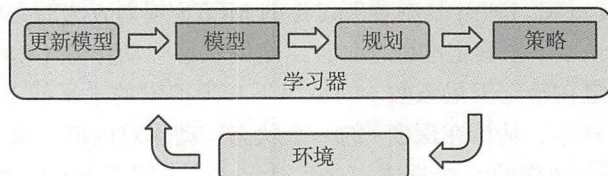


图 4.4 通常, 基于模型的学习器的交错模型顺序地学习和规划, 首先完成对模型的更新, 然后规划在更新的模型上计算一个策略

另外一种方式是在批处理模式下进行模型的更新和规划，仅仅在进行每 k 次动作之后才进行处理 [Deisenroth and Rasmussen, 2011]。然而，这种方法意味着学习器在执行批处理更新时在一些动作之间需要长时间的停顿，这在某些问题上是不可接受的。

DYNA [Sutton, 1990, 1991] 是一种反应性强化学学习结构。在其中，学习器从世界上的一个真实动作或者一个保存的经验开始。不同于值迭代，其中通过在状态空间上迭代对所有状态执行贝尔曼更新，在这里，规划的更新是在随机选择的状态-动作对间进行。算法通过使用贝尔曼方程随机选择的状态-动作来更新动作值，从而更新其策略。以这种方式，实际动作仅需要单个动作值更新，而许多基于模型的更新可以在实际动作间发生。然而，DYNA 框架将模型更新与实际动作循环分开，但它仍然需要许多基于模型的更新，以便使策略相对于其模型成为最优的。

优先扫描 [Moore and Atkeson, 1993] (见第1章) 通过基于优先级选择要更新的状态-动作对来改进 DYNA，而不是随机地选择。它基于其值的预期的改变来按顺序更新状态-动作对。不同于在整个状态空间上进行迭代，优先扫描更新值从模型改变的状态空间向后传播。

118

使用 MCTS 来进行规划而不是使用动态规划技术 (比如值迭代或者策略迭代) 会导致稍微不同的架构。使用动态规划技术，在模型改变后计算整个状态空间的价值函数。一旦计算，不再需要执行动态规划技术，除非模型改变。基于样本的 MCTS 方法将计算集中在学习器可能很快访问的状态，规划滚动必须发生的每一步以向未来的状态靠近。这些算法通常是任意时间算法，因此如果需要的话，可以限制每个步骤的给定计算量。

DYNA-2 是一个将 DYNA 的价值函数更新的理念扩展到使用基于样本规划 [Sliver et al, 2008] 的架构。DYNA-2 维护着单独的线性函数近似，以表示价值函数的永久性和短暂性的记忆。永久记忆是通过在世界中的真实经验进行更新。在每个动作之间，短暂记忆是通过运行 UCT 对学习器的模型进行更新。短暂记忆关注状态空间的较窄的部分 (其中 UCT 正在采样)，用于增大全局价值函数。基于永久性和短暂性价值函数的组合来选择动作。这种架构将粗略的永久性全局价值函数与由 UCT 规划每个步骤创建的更精细的短暂性局部价值函数相结合。

对于一些问题，比如控制机器人或其他物理设备，它可能需要一个实时的架构。在我们上面描述的架构中，模型的更新或规划步骤需要大量的时间。如图 4.5 所示，[Hester et al, 2011] 开发了基于实时模型的强化学习的线性架构，把模型学习、规划和动作分在三个并行线程中。这些线程通过四个共享的数据结构进行通信：要添加进模型的经验列表、规划器使用的模型的副本、规划器规划的当前状态以及学习器的策略。模型学习线程运行在一个循环中，从新经验列表中删除经验，根据这些经验更新其模型，然后拷贝模型到一个版本由规划器使用。规划线程运行一个基于采样的规划算法 (比如 UCT)，从学习器的当前状态进行规划以及更新策略。动作线程添加最近的经验到更新列表中，设置学习器的当前状态用于规划，并从其策略中返回对该状态的最佳动作。动作线程能够在需要的任何频率下立即返回所需的动作。根据动作周期和更新模型的时间长度，模型学习线程每次可以等待新的经验，或者一次更新许多经验。当模型一次合并了许多经验时，算法的表现类似于批处理的学习方法，然而，学习器会在批处理更新发生时继续采取动作[⊖]。

119

⊖ 这个架构的源代码可在此下载：http://www.ros.org/wiki/rl_agent。

到达一个未知的状态，它将执行平衡漂移（采取从该状态选择的动作最少）。如果达到一个已知的状态，它将试图去规划一个尽可能快地获取一个未知状态的探索策略。如果达到一个未知状态的概率大于 $\varepsilon/(2R_{\max})$ ，则继续这个策略。如果没有超过这个范围，该作者证明，规划最优策略必将达到最优策略 ε 范围内的策略。因此，如果达到未知状态的概率没有达到界限，该算法将会规划一个近似的策略并遵循它。当概率不小于 $1-\delta$ ， E^3 算法将在 N 、 T 、 R_{\max} 、 $\frac{1}{\varepsilon}$ 、 $\frac{1}{\delta}$ 的多项式步骤中获得大于最佳值 ε 的返回。

R-MAX [Brafman and Tennenholtz, 2001] 是一个类似的基于模型的算法，其样本效率具有已证明的界限。它也学习一个最大似然模型的列表并跟踪已知、未知的状态。R-MAX 算法在针对不确定性方面比较有效。该算法在它的模型中以到一个吸收状态的转换去替代未知的转换。一个吸收状态即在这个状态中全部的动作离开学习器并且为学习器提供一个域中最大的回报 R_{\max} 。对学习器做如此的激励去探索全部的状态-动作对 m 次，允许它去学习一个准确的模型，从而获得一个最优策略。该算法比 E^3 算法简单，它采用了一种隐式的探索或开发方式而不是 E^3 算法那样的直接、明确的决策。当概率不小于 $1-\delta$ ，R-MAX 算法将会在 N 、 T 、 R_{\max} 、 $\frac{1}{\varepsilon}$ 、 $\frac{1}{\delta}$ 的多项式步骤中获得一个预期的返回值 2ε ，算法 11 是 R-MAX 算法的伪代码。^①

121

```

1: // Initialize  $s_r$  as absorbing state with reward  $R_{\max}$ 
2: for all  $a \in A$  do
3:    $R(s_r, a) \leftarrow R_{\max}$ 
4:    $T(s_r, a, s_r) \leftarrow 1$ 
5: Initialize  $s$  to a starting state in the MDP
6: loop
7:   Choose  $a = \pi(s)$ 
8:   Take action  $a$ , observe  $r, s'$ 
9:   // Update model
10:  Increment  $C(s, a, s')$ 
11:  Increment  $C(s, a)$ 
12:   $RSUM(s, a) \leftarrow RSUM(s, a) + r$ 
13:  if  $C(s, a) \geq m$  then
14:    // Known state, update model using experience counts
15:     $R(s, a) \leftarrow RSUM(s, a) / C(s, a)$ 
16:    for all  $s' \in C(s, a, \cdot)$  do
17:       $T(s, a, s') \leftarrow C(s, a, s') / C(s, a)$ 
18:  else
19:    // Unknown state, set optimistic model transition to absorbing state
20:     $R(s, a) \leftarrow R_{\max}$ 
21:     $T(s, a, s_r) \leftarrow 1$ 
22:  // Plan policy on updated model
23:  Call VALUE-ITERATION
24:   $s \leftarrow s'$ 

```

算法 11 R-Max 输入 S, A, m, R_{\max}

R-MAX 算法所使用的原则在学习框架中形式化，叫作 Knows What It Knows (KWIK) [Li et al, 2008]。对于适合 KWIK 框架的模型学习方法，当查询指定的状态-动作对时，它必须总是做出一个准确的预测或者回答“我不知道”并请求该样例的标签。在强化学习的设定中，KWIK 算法可用作模型学习方法，能够驱动学习器去探索报告“我不知道”这样的状

^① R-MAX 的源码可在：http://www.ros.org/wiki/rl_agent 下载。

态来快速改进模型。KWIK 算法的不足是其常常需要大量的经验去保证当没有报告“我不知道”的时候的准确预测。

4.6 分解域

很多问题可以利用一个分解状态来表示，该状态由一个包含 n 个状态特征的向量表示：

$$s = \langle x_0, \dots, x_n \rangle$$

[122]

比如，一个学习器学习控制一个机器人可以表示成对每个关节的独立的状态特征。在很多情况下，在分解域中的转换概率可以由动态贝叶斯网络（DBN）进行测定。在 DBN 模型中，下一个状态的每一个状态特征可能仅仅依赖于前一个状态和动作的特征的一些子集。一个给定的状态特征所依赖的特征叫作它的父辈。任意一个状态特征的父辈的最大数目叫作 DBN 的最大入度。当使用一个 DBN 转换模型时，假定每一个特征转换都是独立的。这些独立的转换概率可以通过下面的公式融合进整个状态转换的预测中，公式如下：

$$P(s'|s,a) = T(s,a,s') = \prod_{i=0}^n P(x_i|s,a)$$

学习 DBN 转换模型的结构称为结构学习问题。一旦学习到 DBN 的结构，每条边的条件概率必须被学习到。通常，这些概率存储在一个条件概率表格中，即 CPT。

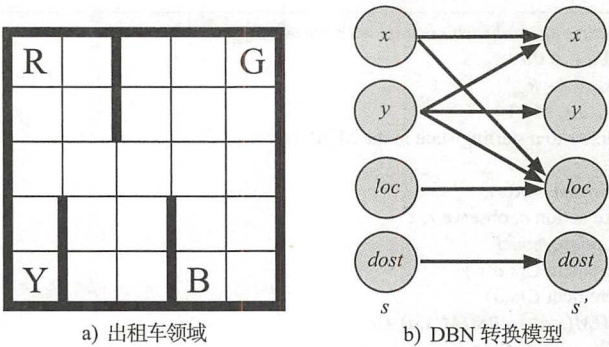


图 4.6 a) 展示了出租车领域；b) 展示了该领域的 DBN 转换模型。在状态 s 中特征 x 只依赖于在状态 s 和特征 y ，特征 y 只依赖于之前的 y 。乘客的目的地只依赖于之前的目的地，当前的位置依赖于之前的一站，就像前一站的出租车的位置 x 和 y

图 4.6 展示了一个在出租车领域的 DBN 示例 [Dietterich, 1998]。在这里，学习器的状态由 4 个特征组成：它的 x 、 y 坐标，乘客的位置，乘客的目的地。学习器的目标是驾驶汽车去接乘客，接上乘客，送达目的地，放下乘客。出租车的 y 坐标仅仅取决于先前 y 坐标位置，而不是取决于 x 坐标或乘客当前位置或乘客的目的地。由于域中的垂直墙，出租车的 x 坐标由 x 和 y 坐标共同决定。如果这个结构是已知的，那么模型学习问题将变得更简单， x 、 y 变量转换的同一个模型可以用于任一乘客位置和目的地的值。

[123]

针对分解域的基于模型的强化学习方法在给予学习器的信息和假设的量方面有所不同。多数假定 DBN 转换模型，状态特征进行独立的转换。一些方法从无需模型的知识开始，必须首先学习 DBN 的结构，然后学习概率。另外一些方法是给出结构，在 DBN 中必须简单地学习与每条边相关联的概率。我们将讨论以下几个变化。

DBN- E^3 算法 [Kearns and Koller, 1999] 将 E^3 扩展到分解域中，其中 DBN 模型的结构是

已知的。在给出 DBN 结构的情况下, 该算法必须学习 DBN 中关联每条边的概率。该算法能够在 DBN-MDP 大量参数的多项动作中学习一个次优的策略, 动作数目将会比所有状态数要小指数级。

类似于 E^3 扩展到 DBN- E^3 , 针对分解域, R-MAX 能够扩展到 FACTORED-R-MAX, 用于给出 DBN 转换模型结构的分解域 [Guestrin et al, 2002]。该算法实现了和 DBN- E^3 相同的样本复杂性界限, 同时, 也保留了 R-MAX 超过 E^3 的实现和简单的优点。

Structure Learning Factored R-MAX (SLF-R-MAX) [Strehl et al, 2007] 将 R-MAX 类型方法应用于 DBN 的结构未知的分解域。当给出 DBN 的最大入度的时候, 它学习 DBN 的结构和条件概率。算法枚举输入要素的所有可能组合作为元素, 然后创建计数器去测量哪个元素是相关的。对于查询状态找到相关元素时, 算法进行预测; 如果没有找到相关元素, 状态即视为未知的。与 R-MAX 算法类似, 该算法给出了在值迭代中对未知状态 R_{\max} 的一个回报, 激励学习器去探索它们。算法的样本复杂度高度依赖于 DBN 的最大入度 D 。因为概率不小于 $1-\delta$, SLF-R-MAX 算法的策略是 ε 最优的, 除了最多 k 步, 其中:

$$k = O\left(\frac{n^{3+2D} AD \ln\left(\frac{nA}{\delta}\right) \ln\left(\frac{1}{\delta}\right) \ln\left(\frac{1}{\varepsilon(1-\gamma)}\right)}{\varepsilon^3 (1-\gamma)^6}\right)$$

n 是域中因素的数量, D 是 DBN 的最大入度, γ 是阻尼系数,

[Diuk et al, 2009] 通过 k-Meteorologists R-MAX (MET-R-MAX) 算法改进了 SLF-R-MAX 的样本复杂度, 其通过引进更高效的算法以选择哪个输入特征对预测是相关的。基于不同的 DBN, 利用预测器的均方误差来提高效率。在学习 DBN 的结构中, 样本复杂度从 $O(n^{2D})$ 提升到了 $O(n^D)$ 。用于该算法的总样本复杂度是用于分解域的已知的最好的约束。

[Chakraborty and Stone, 2011] 提供了一种类似的方法, 其不需要 DBN 的入度信息, 其叫作 Learning Structure and Exploit with R-MAX (LSE-R-MAX)。与 MET-R-MAX 相比, 它使用一个替代线路去解决结构学习问题, 通过假定规划范围信息是满足特定条件的, 而不是使用入度知识。通过这种假设, 在样本复杂度边界上, 与 MET-R-MAX 相比, LSE-R-MAX 解决了结构学习问题, 同时, 在两个测试域中表现得更好。 124

决策树是解决结构学习问题的另外一个方法。通过信息增益可以自然地学习到问题的结构, 确定哪个特征对分裂做出预测是有用的。此外, 它比严格的 DBN 模型更加的概括、泛化。即使对于一个给定特征的父辈状态特征, 决策树也能够决定状态空间的一部分, 特征是不相关联的。比如, 在出租车中, 当载客时, 乘客的位置仅仅依赖于出租车的位置。在其他所有情况下, 预测乘客的位置时都是忽略了出租车的当前位置。

在 SPITI 算法中, 将决策树用于分解域中的学习模型 [Degris et al, 2006]。该算法学习一棵决策树去预测域中的每一个状态特征。该算法在模型上使用结构化的值迭代 [Boutilier et al, 2000] 规划模型, 并且使用 ε -贪婪搜索。比起许多方法在实际应用中使用表格或者 DBN 模型, 该模型的泛化使得它有更好的样本效率。但是, 不能保证决策树能够完整地学习正确的转换模型, 因此, 没有理论证明它的样本高效性。

RL-DT 是另外一种在模型中使用决策树的方法, 它通过使用不同的探索策略以及对状态的相对转换进行建模来改进 SPITI 算法 [Hester and Stone, 2009]。通过预测每一个特征的相对改变, 而不是它的绝对值, 树模型可以对未知状态的转换动态做出更好的预测。此外,

该算法使用一个更加定向的探索策略，在 R-MAX 类型探索之后，将学习器驱动到具有很少被访问的状态，直到学习器找到具有接近 R_{\max} 回报的状态，此时，切换使用模型开发策略。这个算法在网格世界任务中被证明是高效的，比如出租车问题、人型机器人学习点球得分 [Hester et al, 2010]。

图 4.7 展示了一个示例决策树，在一个给定的网格域中，预测学习器的 x 变量的相对变化。决策树能够在学习器的动作和状态上进行拆分，允许它将状态空间拆分成转换动态相同的区域。树的每一片叶子都可以根据叶子的经验结果的比率做出概率预测。网格被着色以匹配树的左边叶子，从而预测学习器何时采取向东的动作。学习器在 MDP 中动作时，树也在线同时构建。开始时，树为空，随后慢慢填充。首先，树会做出状态空间的大部分的预测，比如东或西的动作做什么，之后，慢慢地为各个状态填充叶子，在那里，转换动态是不同于全局动态的。

[125]

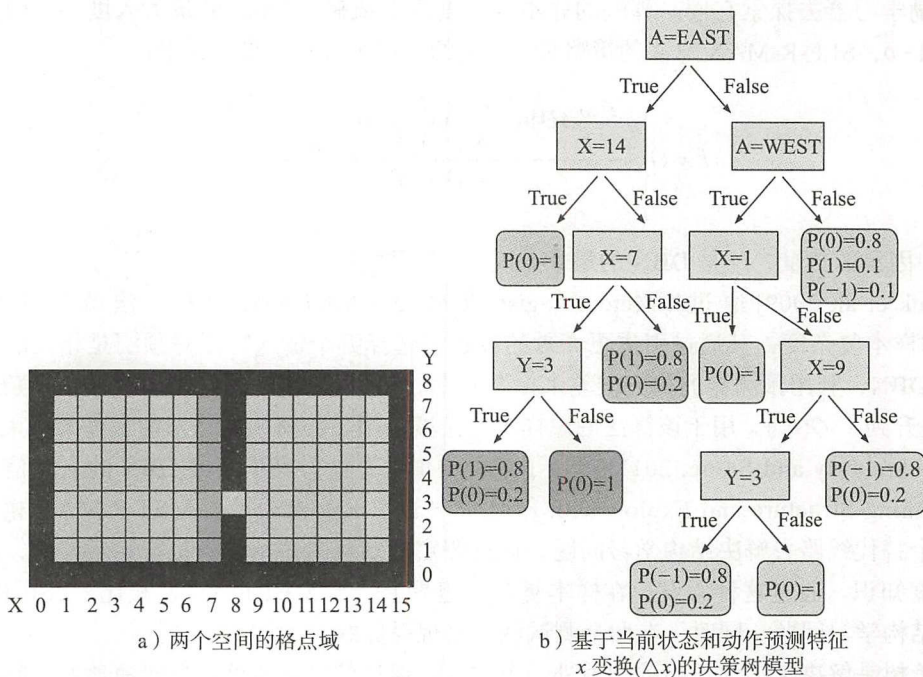


图 4.7 展示了用于预测特征 x 变换的决策树模型。这两个空间的着色对应着树的左侧叶子，其中，学习器采用了 *east* 动作。每个矩形代表树的一个分支，每个圆角矩形代表树的一片叶子，展示了在给定 Δx 值时的概率。例如，如果动作是 *east*，并且 $x = 14$ ，将落到左侧的深灰色叶子处，此处， $\Delta x = 0$ 的概率为 1

4.7 探索

E^3 和 R-MAX 算法的一个核心内容是：在模型中，学习器怎样和什么时候做出搜索性动作（次优），而不是利用模型知道的。基于模型方法的优点之一是它们允许学习器定向的进行探索、规划多步探索策略，而不是使用简单的 ϵ -贪婪或者 softmax 探索这种无模型方法的方式。 E^3 和 R-MAX 算法就是这样做的，通过跟踪每一个状态的访问次数，驱动学习器探索低于一个给定的访问次数的所有状态。然而，如果学习器可以测量模型中的不确定性，那

么它可以驱动探索而不依赖于访问次数。本节将讲述这种方法，主要包括两个方面：1) 它们如何测量模型中的不准确性；2) 它们如何准确地使用不准确性进行探索。

126

基于模型的贝叶斯强化学习方法(第11章)试图在可能的模型上维持一个后验分布来解决探索问题。这种方法是希望解决探索问题的，因为它提供了一个原则方式去跟踪在模型不同部分的学习器不确定性。此外，有了这个明确的不确定性测度，贝叶斯方法可以探索有潜力提供未来回报的状态，而不是仅仅探索状态以减少不确定性。

[Duff, 2003] 提出了一个“最优化探索”去解决探索问题，使用一个增强的状态空间，包括学习器的状态和模型中它自身的信度(称为信度状态 MDP)。学习器的模型包括一个动作如何影响它的状态，在模型上它怎样影响了学习器的信度(以及它相信最有可能的模型)。通过规划更大的扩增的状态空间，学习器就可以优化地进行探索。它知道哪些动作会通过重要且潜在有用的方式改变其模型信度，并可以忽视那些没有用的、仅仅影响部分模型的操作。虽然这种方法本来就是相当高效时，但是在这个扩增的状态空间上进行规划将带来昂贵的计算开销。[Wang et al, 2005] 通过将其与类似 MCTS 的规划结合使得这种方法的计算更加可行。它在整个状态空间上的规划将更加高效，在几次动作后，信度空间的整个部分可以被忽视。BEETLE [Poupart et al, 2006] 通过参数化模型，将模型参数结合在一起降低模型学习问题的规模，这样来使得该方法更加计算可行。然而，这种方法对于多于一个状态的任何问题都是不切实际的。

其他的贝叶斯方法使用模型分布去驱动探索，而不必在状态空间上通过增强模型信度来规划。贝叶斯 DP [Strens, 2000] 和最佳采样集 (BOSS) [Asmuth et al, 2009] 通过从模型的分布中采样并以不同方式使用这些样本来处理探索问题。

贝叶斯 DP 从分布中采样单个模型并进行规划策略，接着，在采样出一个新模型之前，遵循该策略规划多个步骤。在采样新模型之间，学习器会遵循与采样模型一致的策略，根据采样的模型，它可能更加具有探索性或利用开发性。

另一方面，BOSS 在模型的某些部分收集足够的数据时，从模型后验分布采样 k 个模型。然后它将模型合并为具有相同状态空间的单个乐观模型，而不是 kA 个动作的增强动作空间。基本上，由每 A 个动作的 k 个模型的每个预测建模动作。在这个乐观的模型上进行规划驱使学习器每一个状态从 k 个采样模型中任选一个动作。驱动学习器以探索状态空间区域，其中模型是不确定的，因为由于状态空间的该部分中的模型分布的方差，至少一个采样模型可能是乐观的。

127

基于模型的贝叶斯探索 (MBBE) 是由 [Dearden et al, 1999] 提出的一种类似的方法。在模型参数和样本上维持一个分布，并求解 k 个模型去得到一个在动作-值上的分布。它们使用动作-值上的分布去计算完美信息的值 (VPI)。使用 VPI 值以获得高信息增益的行为提供的奖励价值。

这三种方法(贝叶斯 DP、BOSS、MBBE)提供了三种不同的从贝叶斯分布的模型中采样来解决探索问题的方式。这三种方法提供高效的探索的同时，需要学习器在模型上维持一个贝叶斯分布和从分布中采样的模型。同时，它也需要用户事先创建一个定义良好的模型。

[Kolter and Ng, 2009] 试图通过一种叫作贝叶斯探索奖励 (BEB) 的算法将 E^3 和 R-MAX 的多项式收敛证明扩展到贝叶斯方法。通过一些多项式步骤，BEB 遵循贝叶斯最优策略，使用类似于 R-MAX 的探索回报去驱动学习器接近贝叶斯最优策略。BEB 的一个缺点是它需要从狄利克雷分布中提取转换函数。

基于区间估计的模型 (MBIE) [Wiering and Schmidhuber, 1998; Strehl and Littman, 2005] 是一种类似的方法, 它关注于转换概率的分布, 而不是动作 - 值的分布。该算法维持着转换概率的统计置信区间, 其中已采样的转换在相同平均值周围具有更紧密的分布。当选择动作时, 算法通过在计算的置信区间内而且会得到最高策略值的转换概率来计算价值函数。MBIE 十分高效地求解了个别动作最大值之外的最大可能的转换概率。

SLF-R-MAX、MET-R-MAX 和 LSE-R-MAX 在分解域中执行定向的探索 [Strehl et al, 2007; Diuk et al, 2009; Chakraborty and Stone, 2011]。它们使用一个 R-MAX 类型的探索奖励去探索以确定 DBN 转换模型的结构并确定条件概率。它们的探索开销少于 R-MAX 的方法, 因为它们 DBN 模型能够去确定与特定特征预测不相关的某些特征。由于更少的相关特征, 相关特征不同的状态空间比假设每一个状态有不同转换动态时要更小。

根据表格模型, 很显然, 学习器必须探索每一个状态 - 动作对去学习一个准确的模型。这个任务可以通过随机探索来完成, 但是, 使用类似 R-MAX 方法, 定向地探索未访问的状态会更加高效。然而, 当处理比如决策树、DBN 这样的模型, 我们不想学习器访问每一个状态。这些模型的好处之一是它们可以对未知的状态 - 动作对做出合理的预测。因此, 不用探索每一个状态 - 动作对, 学习器使用类似贝叶斯方法 (比如上面提及的 BOSS), 这将更加高效。

TEXPLORE 算法 [Hester and Stone, 2010] 是试图实现这一目标、针对分解域的一种基于模型的方法。以一种随机森林的方式 [Breiman, 2001], 它学习域的多个可能的决策树模型。图 4.8 展示了随机森林模型, 每一个随机森林模型是由 m 个可能的域的决策树模型组成, 它在学习器经验的一个子集上进行训练。学习器在这 m 个模型的均值上进行规划, 同时也能在模型预测的方差上包含一个探索奖励。

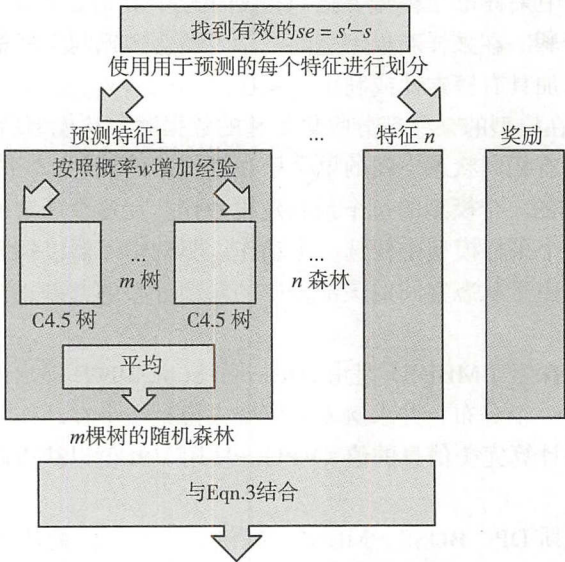


图 4.8 TEXPLORE 的模型学习。该图展示了 TEXPLORE 算法 [Hester and Stone, 2010] 如何学习领域的模型。学习器计算是 s' 和 s 的差分作为转换的影响。然后, 划分状态向量并学习一个随机森林来预测在每个状态特征下的变化。每个随机森林由随机决策树形成, 以概率 w 来更新每个新的经验。随机森林通过平均每棵树的预测来进行预测, 然后, 每个特征的预测组合成该域的一个完整的模型

与 BOSS 或者 MBBE 的多个采样模型类似, TEXPLORE 学习器建立了域的转换模型的 k 个假设。TEXPLORE 以与众不同的方式创建预测下一个状态分布的合并模型, 这些分布是每个模型预测的分布的平均值。在这平均分布上进行规划驱使学习器去探索有前途的转换, 当它们的概率足够高时 (当许多模型预测有前途的结果, 其中一个预测具有很高的概率)。这与 BOSS 的方法类似, 但考虑了预测乐观结果的采样模型的数量。在 TEXPLORE 中, 一个给定模型的预测具有概率 $\frac{1}{k}$, 而 BOSS 创建的额外动作总是假定为特定模型预测的转换。 [129]

在示例中, 这个差异变得十分明显。如果 TEXPLORE 的模型不一致并且平均模型预测出现特定负面结果的机会很小, 学习器将基于它可能发生的机会来避免它。然而, BOSS 学习器将简单地从不同的模型中选择一个动作, 并忽略这些负回报的可能性。另一方面, 假设 TEXPLORE 的平均模型预测到发生高价值结果的概率, 如果结果的价值相对于其概率而言很高, 这可能值得探讨。TEXPLORE 已经在基于 300 000 个状态的网格世界中使用 [Hester and Stone, 2010], 并且在自动车辆领域实时运行 [Hester et al, 2011]^①。

[Schmidhuber, 1991] 试图将学习器推向模型改进最多的地方, 而不是试图估计模型在哪里最差。作者采用传统的基于模型的强化学习方法, 并且添加了一个置信模块, 训练该模块以预测模型误差的绝对值。该模块可以用于创建内在回报, 鼓励学习器去探索高错误的状态-动作对, 但是除了不良的模型之外, 学习器将被吸引到噪声状态。相反, 作者增加了另一个被训练去置信模块输出变化的模块。使用这个模块, 驱动学习器探索状态空间中最能改进模型的预测误差的部分。

[Baranes and Oudeyer, 2009] 提出了一个基于类似想法的算法, 叫作鲁棒的智能适应好奇心算法 (R-IAC), 一种提供内在回报去激励发展中的学习器探索的方法。它们的方法不采用强化学习框架, 但是在许多方面是类似的。在其中, 它们将状态空间划分为区域, 并在每一个区域中学习转换动态模型。它们维持对每一个区域预测误差的估计, 并且使用该误差的梯度作为学习器的内在回报, 驱动学习器去探索预测误差改进最大的区域。由于这种方法没有使用强化学习框架, 它们的算法选择动作只是最大化即时回报, 而不是未来奖励的折扣和。它们的方法没有结合外部奖励, 但是它们可以用于向现有的强化学习学习器提供内在奖励。

4.8 连续域

在本章中, 算法大部分的描述都假定学习器是在离散状态空间中操作。然而, 许多现实世界的问题 (比如机器人控制) 涉及连续的状态和动作。这些方法可以通过量化状态空间来扩展到连续问题, 但是非常精细的离散导致大量的状态, 以及在离散化中一些信息的丢失。基于无模型的方法通过近似函数的使用, 可以相当容易地扩展到连续域。然而, 基于模型的方法扩展到连续域存在多个挑战: 1) 学习连续性模型; 2) 在连续性状态空间上规划; 3) 探索一个连续性状态空间。连续性的方法在第 7 章中进一步描述。 [130]

学习连续域的模型需要下一个状态的预测和来自连续状态的奖励。与离散情况不同, 不能简单地学习用于一些离散状态集合的表格模型。必须使用某种形式的函数近似, 因为许多真实值状态可能永远不会被访问。常见的方法是使用回归或基于实例的技术来学习一个连续性模型。

一个更困难的问题是在连续状态空间上进行规划。学习器需要从无限数量的状态中知道

① TEXPLORE 算法源码的下载地址为 http://www.ros.org/wiki_agent。

最佳的动作。同样，这个可以通过对策略的某种形式的函数近似来完成，或者状态空间可以为规划目的而离散化（即使用于学习模型）。此外，第 7 章讨论了许多用于连续状态空间的无模型方法，比如策略梯度方法 [Sutton et al, 1999] 或具有函数近似的学习，这些可用于规划模型的策略。

拟合值迭代（FVI）[Gordon, 1995] 将值迭代适配适应连续状态空间。从无限状态空间中采样出有限的状态集合的值进行更新和迭代，然后，对它们的值拟合函数逼近。如果函数逼近符合一些收缩标准，则拟合值迭代证明可以收敛。

用于连续状态空间的最早的基于模型强化学习算法之一是 PARTI-GAME 算法 [Moore and Atkeson, 1995]。它不工作在典型的强化学习框架下，具有确定性的动态和一个目标区域，而不是一个回报函数。该算法用于学习和规划状态空间的离散化，自适应地增加其在状态空间的感兴趣部分中的分辨率。当规划器无法找到对目标的策略时，成功和失败的单元边界上的单元格将被进一步分割，以提高离散化的分辨率。这种方法促使学习器在需要时具有一个更确定的动态模型，并在其他地方具有一个更一般性的模型。

与 PARTI-GAME 算法的分区方法不同，[Ornstein and Sen, 2002] 在他们的基于内核的强化学习算法中使用了基于实例的模型。该算法保存了其经历的所有转换。当对所查询状态-动作进行预测时，模型使用内核函数的加权，基于附近转换的均值进行预测。这个模型与近似动态规划结合，创建一个完整的基于模型的方法。

[Deisenroth and Rasmussen, 2011] 使用高斯过程（GP）回归在它们的学习控制的概率推理（PILCO）模型的算法中学习模型。GP 回归模型推广到未知状态，并为其预测提供了置信界限。学习器规划假设下一个状态分布匹配置信界限，当来自于下一个状态分布的某些状态被高度评估时鼓励学习器探索。该算法还使用 GP 回归来表示其策略，并且使用策略迭代来计算策略。它以批处理模式运行，或在现实世界中采取批量操作，然后重新计算它的模型和策略。该算法学习控制具有少量样本的物理车杆装置，但是在每 2.5 秒的动作之后暂停 10 分钟的计算。

[Jong and Stone, 2007] 提出了一种称为 FITTED-R-MAX 的算法，使用基于实例的模型扩展 R-MAX 到连续域。当查询一个状态-动作对时，算法使用查询的状态-动作对的最近实例。他们使用最近实例的相对效应来预测所查询的状态-动作的相对改变。他们的算法可以提供下一个状态的分布，然后用于具有拟合值迭代的规划。学习器被鼓励去探索没有足够实例（含有 R-MAX 类型探索奖励）的状态空间部分。

最小二乘策略迭代（LSPI）[Lagoudakis and Parr, 2003] 是一种使用函数近似规划的流行方法（详见第 3 章）。当使用线性函数逼近时，它执行近似策略迭代。它针对给定的经验集，计算来自贝尔曼方程的最小化的最小二乘差分的策略参数。这些经验可以来自于生成模型或学习器保存的经验。然而，由于昂贵的计算，LSPI 通常用于通过随机游走收集的经验批量学习。[Li et al, 2009] 通过类似于 FITTED-R-MAX 的探索奖励来扩展 LSPI 以执行在线探索。

[Nouri and Littman, 2010] 采用了不同的方法，侧重于在连续域中探索。他们提出了一个称为 Dimension Reduction in Exploration（DRE）的算法，该算法在学习转换函数中使用降维的方法，该转换函数自动地发现用于预测的相关状态特征。他们独立地预测每个特征，并且使用一个来自模型的“已知”标准来驱动探索。他们结合这个模型和拟合值迭代来规划每一个步骤。

虽然强化学习通常关注于离散动作,但是存在许多控制问题需要连续的控制信号。在这种情况下尝试找到最好的动作可能是一个困难的问题。二元动作搜索 [Pazis and Lagoudakis, 2009] 通过离散化动作空间并将连续动作选择问题分解为一系列二进制动作选择来提供对这个问题的可能的解决方案,每一个动作决定要采取连续动作值的一位。或者,可以使用函数逼近的方法(比如,演员-评论家方法)来表示策略,并且适当地更新函数近似以输出最佳连续性动作 [Sutton et al, 1999, van Hasselt and Wiering, 2007]。[Weinstein et al, 2010] 提出了一种称为 HOOT 的不同方法,使用 MCTS 类型搜索来分割和搜索连续动作空间。在连续动作空间中搜索,选择连续空间上越来越小的分区,直到达到搜索树的叶子并选择一个动作。

132

4.9 实证比较

前面已经研究了基于模型的方法,它们怎样结合规划和模型学习,它们如何探索,如何处理连续域,我们现在介绍一些简要的代表性实验来说明它们最重要的特性。虽然基于模型的方法在有限动作机会的大型领域中最有用(见 4.10 节),我们可以说明它们在简单玩具领域的性质。^①

我们在出租车领域中比较 R-MAX (见 4.5 节) 和 Q 学习 (一个典型的无模型方法)。R-MAX 运行所需要的访问次数被认为是已知的状态 M , 设为 5。学习以学习率 0.3 和 $\epsilon=0.1$ 的 ϵ -贪婪探索运行。这两种方法都以 0.99 的阻尼系数运行。如图 4.9a 所示,出租车域 [Dietterich, 1998] 是一个 5×5 的网格,有 4 个标记,标记以下颜色之一:红色、绿色、蓝色和黄色。学习器的状态包括在网格中的 x 、 y 坐标、乘客的位置(地标或出租车)、乘客的目的地(地标)。学习器的目标是驾驶出租车到乘客的位置,载上乘客,送达乘客的目的地,放下乘客。学习器有 6 个动作可以采用。学习器前 4 个动作是向正方形的方向(北,南,西,东)移动,其概率为 0.8,垂直方向的概率为 0.1。如果所得到的方向被阻挡,则学习器停留在其处。第 5 个动作是拾取动作,在出租车位置载起乘客。第 6 个动作是放下动作,放下乘客。每个动作都会产生 -1 的回报,而失败的拾取或放下动作将产生 -10 的回报。片段终止于成功的放下动作,产生一个 +20 的回报。每个片段从乘客的位置开始,目的地随机从 4 个地标中选择,且学习器在网格中的一个随机位置。

图 4.9b 显示了经过平均 30 次试验,学习所累积的平均回报和出租车领域中的 R-MAX 比较。R-MAX 由于它探索了所有“未知”状态,早早收到了大的负回报。然而,这种探索将会比 Q 学习更快得到最优策略。

接下来,我们在车杆平衡任务中展示了一些基于模型方法和 Q 学习的性能。车杆平衡是一个连续任务,如图 4.10a 所示,其中学习器必须保持杆的平衡,同时保持推车在轨道上。学习器有两个动作,对车向任一方向施加 10N 的力。-5N 到 5N 之间的均匀噪声被加到该力上。状态由 4 个特征组成:杆的位置,杆的速度,车的位置,车的速度。学习器每一个步骤都收到 +1 的回报直到片段结束。如果杆落下,车子离开轨道或已经经历 1000 个步骤,则片段结束。这个任务在 50Hz 处模拟。对于离散的方法,我们将 4 个维度的每一维离散成 10 个值,总共 10 000 个状态。

133

① 重建这些实验的源代码可在以下网址下载: http://www.ros.org/wiki/reinforcement_learning。

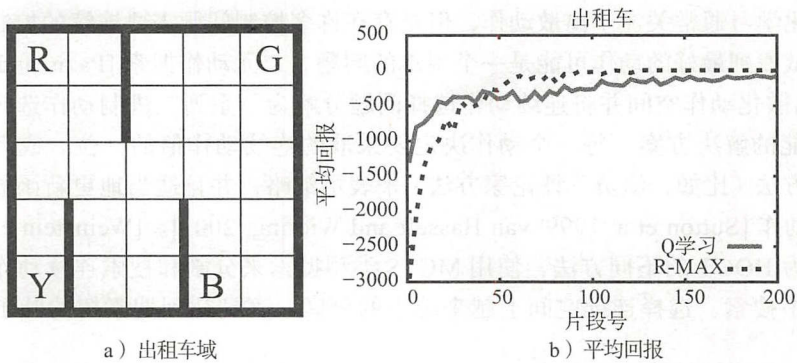


图 4.9 a) 展示了出租车领域；b) 展示了 Q 学习和 R-MAX 在出租车领域的 30 次试验中，每个片段的平均回报

对于无模型方法，我们比较离散域上的 Q 学习和在任务的连续表示上使用 tile 编码的 Q 学习。这两种方法都以 0.3 的学习率和 $\epsilon = 0.1$ 的 ϵ -贪婪探索进行运行。具有 tile 编码的 Q 学习使用 10 个连接 tile，每一个维度可以划分为 4 个 tile。我们也比较了四种基于模型的方法：两种离散方法和两种连续方法。离散方法是 R-MAX [Brafman and Tennenholtz, 2001]，其使用表格模型以及 TEXPLORE 方法 [Hester and Stone, 2010]。TEXPLORE 使用决策树来对转换的相对效应建模和贪婪地模拟一个模型，该模型是多个可能模型的均值。这两种方法都在离散算法上运行。这里再次以 $M = 5$ 运行 R-MAX。使用 $b = 0$, $w = 0.55$, $f = 0.2$ 来运行 TEXPLORE。我们还评估 FITTED-R-MAX [Jong and Stone, 2007]，它是连续域上 R-MAX 的扩展。CONTINUOUS TEXPLORE 是 TEXPLORE 在连续域上的扩展，它使用回归树来模拟连续状态而不是离散的决策树。CONTINUOUS TEXPLORE 使用 TEXPLORE 相同的参数，并以模型宽度为 0.05、分辨率系数为 4 运行 FITTED-R-MAX。

在 4.10b 中展示了平均超过 30 次实验的算法的平均回报。两个版本的 R-MAX 都需要长时间的探索，并且不会得到太多的回报。带有 tile 编码的 Q 学习表现得比离散 Q 学习更好，因为它能够横跨状态间的值来更快地学习。同时，TEXPLORE 方法表现出了卓越的泛化和探索性，因为它们比起其他方法在片段 6 之后，每一片段产生更多的回报。对于每种方法，算法的连续版都比离散版更好。

134

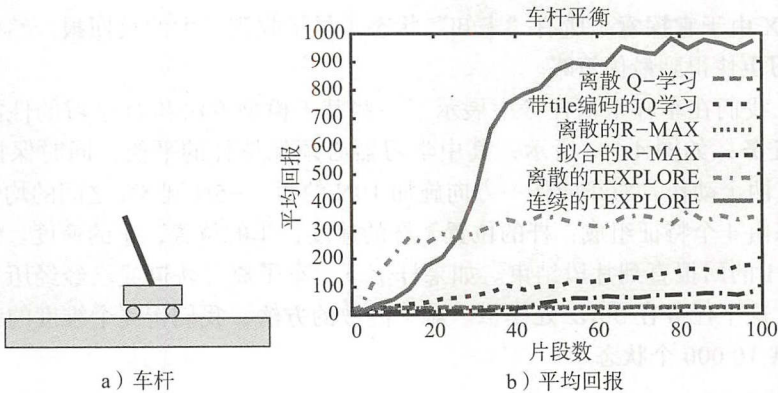


图 4.10 a) 展示了车杆平衡任务。b) 展示了关于车杆平衡的算法在 30 次试验中每个片段的平均回报

这些实验说明了在基于模型和无模型方法间的权衡，以及不同的基于模型的方法间的一些权衡。基于模型的方法（比如 R-MAX）花费很长的时间探索域，以保证收敛到最优策略；其他方法（比如 Q 学习）在探索期间执行得比 R-MAX 更好。另一方面，比如像 TEXPLORE 一类的方法试图学习模型，不像 R-MAX 那样能够快速地学习，但是，既然它们的模型可能不是完全正确的，所以它们不提供相同的保证。此外，当在连续域工作时，算法直接与连续变量一起工作而不需要离散化，这是一个明显的优点。

4.10 扩展

与无模型的强化学习相比，虽然基于模型的强化学习在样本效率方面提供了巨大的前景，但这种优势是以牺牲用于学习模型和规划的更多的计算和空间复杂性为代价的。因此，这些方法的评估常常限制于小型领域，比如山地车、出租车或水坑世界。此外，即使有效的算法也可能在实际任务中采取了太多的动作。将基于模型的强化学习应用到更大更现实的问题的时候，这些问题都必须被解决。

使用分解的模型和泛化，基于模型的强化学习在更大问题上已经实现了样本高效。然而，在这些情况下，当模型改变时，它仍然需要规划整个状态空间，需要大量的计算。文献中提到的解决这个问题的一种方式是将这些方法与基于样本的规划器比如 UCT 结合 [Kocsis and Szepesvári, 2006]。[Walsh et al, 2010] 证明，当使用称为前向搜索稀疏采样 (FSSS) 的基于样本的规划器时，其方法对样本复杂度的限制仍然有效，这是更保守的 UCT 版本。FSSS 维护树中每个节点值的统计上限和下限，并仅寻找最优的子树。

[135]

为了保证找到最优策略，任何基于模型的算法都必须至少访问域中的每个状态-动作对。即使在某些情况下，这种动作的数量很大，无论是因为域很大，还是因为动作是昂贵或危险的。这些界限可以通过假定 DBN 转换模型在分解域中来改进。通过假定提前知道 DBN 的结构，DBN-E³ 和 FACTORED-R-MAX 算法 [Kearns and Koller, 1999; Guestrin et al, 2002] 都能够在多个 DBN-MDP 的参数的多个动作多项式中学习一个次优的策略，其代价指数级地小于整个状态的数量。

这个问题的另一种方法是通过决策树学习 DBN 的结构，如 TEXPLORE 算法 [Hester and Stone, 2010]。这种方法放弃了最优性的保证，因为正确的 DBN 结构可能没有被学到，但它可以通过少量的动作学习许多高回报（如果不是最优）策略。TEXPLORE 算法通过随机森林对 MDP 建模，并探索其模型不确定的地方，但不探索域中的每一个状态-动作对。

提高这种算法的采样效率的另一种方法是将一些人类知识引入学习器。这样做的一种方法是提供人类的经验轨迹，学习器可以使用该轨迹来构建其模型。比如，[Ng et al, 2003] 从专家用户记录的数据中学习遥控直升机的动态模型。然后，他们使用策略搜索强化学习的方法来学习一个策略，用学习到的模型来使得直升机飞行。

在真实世界中，决策任务的另一个问题是它们常常涉及部分可观察状态，其中，学习器不能从其观察到的唯一的标识确认其状态。U-TREE 算法 [McCallum, 1996] 是解决这一问题的一种基于模型的算法。它使用树来学习模型，该树可以将之前的状态和动作结合在树的分裂中，以及当前的状态和动作。历史状态和动作可以用于准确地确定学习器的真实当前状态。然后，算法在这个模型上使用值迭代来规划一个策略。

扩展到更大、更复杂域的另一方法是使用关系模型（详见第 8 章）。这里，世界被表示为一组文字，学习器可以通过类似 STRIPS 规划运算符的形式 [Fikes and Nilsson, 1971] 学

136

习模型。因为这些规划运算符是相互关联的，学习器可以很容易地推广它的动作。[Pasuls et al, 2004] 提出了一种以规则集的形式学习概率关系模型的方法。这些规则描述特定的动作如何影响状态。从一组基于学习器到目前为止所看到的经历的规则开始，对规则集执行搜索以优化分数，从而促成简单、一般的规则。面向对象的强化学习 [Diuk et al, 2008] 根据对象定义世界，对关系型强化学习采取类似的方法。

4.11 总结

基于模型的方法在与环境交互的同时，在线学习 MDP 的模型，然后使用它们的近似模型去计算策略。如果算法能够足够快速地学习准确的模型，基于模型的方法可以比无模型方法更加样本有效。使用学习到的准确的模型，不需要世界上任何额外的经验，就可以规划出最优策略。此外，这些方法可以使用它们的模型规划出多步探索策略，相比无模型方法，使得它们能够执行更加有针对性的探索。

表 4.2 显示了本章描述的基于模型方法的总结。由于 R-MAX [Brafman and Tennenholtz, 2001] 的理论保证和易于使用实现的特点，它是最常用的基于模型的方法之一。MET-R-MAX [Diuk et al, 2009] 和 LSE-R-MAX [Chakraborty and Stone, 2011] 在分解域的样本复杂性边界方面最先进。这里还有其他的方法，在无理论保证的情况下，表现得同样好或更好，比如高斯过程强化学习 [Deisenroth and Rasmussen, 2011; Rasmussen and Kuss, 2004] 或 TEXPLORE [Hester and Stone, 2010]。

基于模型的强化学习当前研究的主要方向是扩展到更大、连续的状态和动作空间的方法。实现这一目标要求算法更加样本高效，依靠更高效率的计算规划器，并开发更好的探索方法。

表 4.2 本章算法的总结

算法	章节	模型类型	关键特征
DHP	4.2	神经网络	在演员 - 评论家框架中使用神经网络
LWR RL	4.2	局部加权回归	生成交叉状态模型的一种方法
DYNA	4.4	表格	在实际动作和保存经验上的贝尔曼更新
优化延伸	4.4	表格	从变化后的状态后向值更新的性能延伸
DYNA-2	4.4	表格	使用 UCT 模拟更新到临时内存
实时结构	4.4	任意	允许实时动作的并行结构
E ³	4.5	表格	搜索或探索的明确决策
R-MAX	4.5	表格	给“未知”状态的回报
DBN-E ³	4.6	为给定的 DBN 模型学习概率	学习 DBN 参数的动作多项式
Factored-R_MAX	4.6	为给定的 DBN 模型学习概率	学习 DBN 参数的动作多项式
SLF-R-MAX	4.6	为所有可能的 DNB 结构学习模型	当模型不确定或者模型不一致时搜索
MET-R-MAX	4.6	有效地学习 DBN 结构和概率	当模型不确定或者模型不一致时搜索
LSE-R-MAX	4.6	不一致时，有效地学习 DBN	当模型不确定或者模型不一致时搜索
SPITI	4.6	决策树	在交叉状态上的模型泛化
RL-DT	4.6	有相关影响的决策树	在交叉状态上的模型泛化，类似 R-MAX 的搜索
优化探测	4.7	保持模型分布	增广信念状态空间上的计划
BEETLE	4.7	保持参数化模型分布	增广信念状态空间上的计划
贝叶斯 DP	4.7	保持模型分布	使用分布采样的模型的计划
BOSS	4.7	保持模型分布	使用从采样模型建立的合并模型的计划

(续)

算法	章节	模型类型	关键特征
MBBE	4.7	保持模型分布	使用动作 - 值到计算 VPI 上的分布
BEB	4.7	保持模型狄利克雷分布	给予贝叶斯策略回报
MBIE	4.7	保持转换概率分布	采用转换概率的最大值作为下一状态
TEXPLORE	4.7	每个特征的随机森林模型	在平均模型上逼近最优策略的计划
固有的好奇心	4.7	监督学习模型	在模型中给提升以固有的回报
R-LAC	4.7	监督学习模型	基于模型的提升选择动作
PILCD	4.8	高斯过程回归	使用下一个状态预测的不确定性的计划
PARTI-GAME	4.8	基于转换动态划分状态空间	非均匀地划分状态空间
基于核的 RL	4.8	使用核距离的基于实例的模型	使用核基于相似状态做出预测
TTED-R-MAX	4.8	基于实例的模型	基于应用相似转换的相互影响进行预测
DRE	4.8	降维技术	在高维状态空间中起作用
U-TREE	4.10	使用历史的树模型	在部分的观察域中做预测
关系型	4.10	关系型规则集	学习类似 STRIPS 的关系操作符

致谢。本次工作由得克萨斯大学奥斯汀分校人工智能实验室的智能体研究组 (LARG) 完成。LARG 研究部分得到国家科学基金会 (IIS-0917122)、ONR (N00014-09-1-0658) 和联邦公路管理局 (DTFH61-07-H-00030) 的资助。

参考文献

- Asmuth, J., Li, L., Littman, M., Nouri, A., Wingate, D.: A Bayesian sampling approach to exploration in reinforcement learning. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI (2009)
- Atkeson, C., Moore, A., Schaal, S.: Locally weighted learning for control. Artificial Intelligence Review 11, 75–113 (1997)
- Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47(2), 235–256 (2002)
- Baranes, A., Oudeyer, P.Y.: R-IAC: Robust Intrinsically Motivated Exploration and Active Learning. IEEE Transactions on Autonomous Mental Development 1(3), 155–169 (2009)
- Boutillier, C., Dearden, R., Goldszmidt, M.: Stochastic dynamic programming with factored representations. Artificial Intelligence 121, 49–107 (2000)
- Brafman, R., Tennenholtz, M.: R-Max - a general polynomial time algorithm for near-optimal reinforcement learning. In: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI), pp. 953–958 (2001)
- Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
- Chakraborty, D., Stone, P.: Structure learning in ergodic factored MDPs without knowledge of the transition function's in-degree. In: Proceedings of the Twenty-Eighth International Conference on Machine Learning, ICML (2011)
- Dearden, R., Friedman, N., Andre, D.: Model based Bayesian exploration. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI), pp. 150–159 (1999)
- Degrís, T., Sigaud, O., Willemin, P.H.: Learning the structure of factored Markov Decision Processes in reinforcement learning problems. In: Proceedings of the Twenty-Third International Conference on Machine Learning (ICML), pp. 257–264 (2006)
- Deisenroth, M., Rasmussen, C.: PILCO: A model-based and data-efficient approach to policy search. In: Proceedings of the Twenty-Eighth International Conference on Machine Learning, ICML (2011)

- Dietterich, T.: The MAXQ method for hierarchical reinforcement learning. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML), pp. 118–126 (1998)
- Diuk, C., Cohen, A., Littman, M.: An object-oriented representation for efficient reinforcement learning. In: Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML), pp. 240–247 (2008)
- Diuk, C., Li, L., Leffler, B.: The adaptive-meteorologists problem and its application to structure learning and feature selection in reinforcement learning. In: Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML), p. 32 (2009)
- Duff, M.: Design for an optimal probe. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML), pp. 131–138 (2003)
- Even-dar, E., Mansour, Y.: Learning rates for q-learning. *Journal of Machine Learning Research*, 1–25 (2001)
- Fikes, R., Nilsson, N.: Strips: A new approach to the application of theorem proving to problem solving. Tech. Rep. 43r, AI Center, SRI International, 333 Ravenswood Ave, Menlo Park, CA 94025, SRI Project 8259 (1971)
- Gordon, G.: Stable function approximation in dynamic programming. In: Proceedings of the Twelfth International Conference on Machine Learning, ICML (1995)
- Guestrin, C., Patrascu, R., Schuurmans, D.: Algorithm-directed exploration for model-based reinforcement learning in factored MDPs. In: Proceedings of the Nineteenth International Conference on Machine Learning (ICML), pp. 235–242 (2002)
- van Hasselt, H., Wiering, M.: Reinforcement learning in continuous action spaces. In: IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL), pp. 272–279 (2007)
- Hester, T., Stone, P.: Generalized model learning for reinforcement learning in factored domains. In: Proceedings of the Eight International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS (2009)
- Hester, T., Stone, P.: Real time targeted exploration in large domains. In: Proceedings of the Ninth International Conference on Development and Learning, ICDL (2010)
- Hester, T., Quinlan, M., Stone, P.: Generalized model learning for reinforcement learning on a humanoid robot. In: Proceedings of the 2010 IEEE International Conference on Robotics and Automation, ICRA (2010)
- Hester, T., Quinlan, M., Stone, P.: A real-time model-based reinforcement learning architecture for robot control. *ArXiv e-prints* 11051749 (2011)
- Jong, N., Stone, P.: Model-based function approximation for reinforcement learning. In: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS (2007)
- Kakade, S.: On the sample complexity of reinforcement learning. PhD thesis, University College London (2003)
- Kearns, M., Koller, D.: Efficient reinforcement learning in factored MDPs. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI), pp. 740–747 (1999)
- Kearns, M., Singh, S.: Near-optimal reinforcement learning in polynomial time. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML), pp. 260–268 (1998)
- Kearns, M., Mansour, Y., Ng, A.: A sparse sampling algorithm for near-optimal planning in large Markov Decision Processes. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI), pp. 1324–1331 (1999)
- Kocsis, L., Szepesvári, C.: Bandit based monte-carlo planning. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) *ECML 2006. LNCS (LNAI)*, vol. 4212, pp. 282–293. Springer, Heidelberg (2006)
- Kolter, J.Z., Ng, A.: Near-Bayesian exploration in polynomial time. In: Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML), pp. 513–520 (2009)
- Lagoudakis, M., Parr, R.: Least-squares policy iteration. *Journal of Machine Learning Research* 4, 1107–1149 (2003)

- Li, L., Littman, M., Walsh, T.: Knows what it knows: a framework for self-aware learning. In: Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML), pp. 568–575 (2008)
- Li, L., Littman, M., Mansley, C.: Online exploration in least-squares policy iteration. In: Proceedings of the Eight International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 733–739 (2009)
- McCallum, A.: Learning to use selective attention and short-term memory in sequential tasks. In: From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior (1996)
- Moore, A., Atkeson, C.: Prioritized sweeping: Reinforcement learning with less data and less real time. *Machine Learning* 13, 103–130 (1993)
- Moore, A., Atkeson, C.: The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning* 21, 199–233 (1995)
- Ng, A., Kim, H.J., Jordan, M., Sastry, S.: Autonomous helicopter flight via reinforcement learning. In: Advances in Neural Information Processing Systems (NIPS), vol. 16 (2003)
- Nouri, A., Littman, M.: Dimension reduction and its application to model-based exploration in continuous spaces. *Mach. Learn.* 81(1), 85–98 (2010)
- Ornstoneit, D., Sen, S.: Kernel-based reinforcement learning. *Machine Learning* 49(2), 161–178 (2002)
- Pasula, H., Zettlemoyer, L., Kaelbling, L.P.: Learning probabilistic relational planning rules. In: Proceedings of the 14th International Conference on Automated Planning and Scheduling, ICAPS (2004)
- Pazis, J., Lagoudakis, M.: Binary action search for learning continuous-action control policies. In: Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML), p. 100 (2009)
- Poupart, P., Vlassis, N., Hoey, J., Regan, K.: An analytic solution to discrete Bayesian reinforcement learning. In: Proceedings of the Twenty-Third International Conference on Machine Learning (s), pp. 697–704 (2006)
- Prokhorov, D., Wunsch, D.: Adaptive critic designs. *IEEE Transactions on Neural Networks* 8, 997–1007 (1997)
- Rasmussen, C., Kuss, M.: Gaussian processes in reinforcement learning. In: Advances in Neural Information Processing Systems (NIPS), vol. 16 (2004)
- Schaal, S., Atkeson, C.: Robot juggling: implementation of memory-based learning. *IEEE Control Systems Magazine* 14(1), 57–71 (1994)
- Schmidhuber, J.: Curious model-building control systems. In: Proceedings of the International Joint Conference on Neural Networks, pp. 1458–1463. IEEE (1991)
- Silver, D., Sutton, R., Müller, M.: Sample-based learning and search with permanent and transient memories. In: Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML), pp. 968–975 (2008)
- Strehl, A., Littman, M.: A theoretical analysis of model-based interval estimation. In: Proceedings of the Twenty-Second International Conference on Machine Learning (ICML), pp. 856–863 (2005)
- Strehl, A., Diuk, C., Littman, M.: Efficient structure learning in factored-state MDPs. In: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, pp. 645–650 (2007)
- Strens, M.: A Bayesian framework for reinforcement learning. In: Proceedings of the Seventeenth International Conference on Machine Learning (ICML), pp. 943–950 (2000)
- Sutton, R.: Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In: Proceedings of the Seventh International Conference on Machine Learning (ICML), pp. 216–224 (1990)
- Sutton, R.: Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bulletin* 2(4), 160–163 (1991)
- Sutton, R., McAllester, D., Singh, S., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: Advances in Neural Information Processing Systems (NIPS), vol. 12, pp. 1057–1063 (1999)

- Venayagamoorthy, G., Harley, R., Wunsch, D.: Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator. *IEEE Transactions on Neural Networks* 13(3), 764–773 (2002)
- Walsh, T., Goschin, S., Littman, M.: Integrating sample-based planning and model-based reinforcement learning. In: *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence* (2010)
- Wang, T., Lizotte, D., Bowling, M., Schuurmans, D.: Bayesian sparse sampling for on-line reward optimization. In: *Proceedings of the Twenty-Second International Conference on Machine Learning (ICML)*, pp. 956–963 (2005)
- Wang, Y., Gelly, S.: Modifications of UCT and sequence-like simulations for Monte-Carlo Go. In: *IEEE Symposium on Computational Intelligence and Games* (2007)
- Weinstein, A., Mansley, C., Littman, M.: Sample-based planning for continuous action Markov Decision Processes. In: *ICML 2010 Workshop on Reinforcement Learning and Search in Very Large Spaces* (2010)
- Wiering, M., Schmidhuber, J.: Efficient model-based exploration. In: *From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior*, pp. 223–228. MIT Press, Cambridge (1998)

强化学习中的迁移：框架和概观

Alessandro Lazaric

摘要

强化学习中的迁移是一个新的研究领域，重点研究从一组源任务传递知识到目标任务的方法。每当任务相似时，所迁移的知识可以由学习算法用来解决目标任务，并显著地提高其性能（比如，通过减少所需要的样本数去获得一个次优的性能）。在本章中，我们提供一般迁移问题的形式化，归纳总结迄今为止已经研究的主要设置，并且回顾强化学习中迁移的最重要的方法。

5.1 简介

迁移知识的思想起源于心理学和认知科学，它通过把不同但相关的任务进行传递，以改进机器学习（ML）算法的性能。许多心理学研究（比如，参见 [Thorndike and Woodworth, 1901; Perkins et al, 1992] 表明，人类可以通过传递解决类似任务而保留的知识来更好更快地学习一个任务。机器学习中的迁移以设计迁移方法为目的，其分析从一组源任务中收集的知识（比如，样本、解决方案），并迁移它，以便将目标任务的学习过程偏向一组好的假设。如果迁移方法成功地识别了源和目标任务间的相似性，则迁移的知识可以提高目标任务的学习性能。保留和重用知识以改进学习算法的想法可以追溯到机器学习的早期阶段。实际上，广泛认识到，一个好的表示是任何学习算法最关键的方面，并且根据手头的任务自动地改变表示的技术的发展是机器学习中大部分研究的主要目标之一。迁移学习的大多数研究 [Fawcett et al, 1994] 将 ML 中通常采用的单一问题视角定义为对良好表征归纳构建的有效方法定义的限制。另一方面，从心理学和神经科学研究中获得灵感 [Gentner et al, 2003; Gick and Holyoak, 1983]。考虑迁移的观点，其中假定学习任务是相关的并且知识被保留和迁移，这被认为是进行设计有效归纳偏向技术的最适合的视角 [Utgoff, 1986]。

143

强化学习中的迁移。迁移算法已经成功地改进了许多监督学习问题中的学习算法的性能，比如推荐系统、医疗决策、文本分类和一般游戏之类。近年来，迁移的研究也集中在强化学习范例和强化学习算法如何从知识迁移中受益。原则上，传统的强化学习已经提供了机制来为任何任务学习解决方案，而不需要人的监督。尽管如此，学习一个几乎最优的解决方案所需的样本数量在现实世界问题中往往令人不满意的，除非有来自领域专家的先验知识可以使用。此外，每当手中的任务改变时，学习过程必须从头重新开始，即使类似的问题已经得到了解决。迁移算法从解决一组类似的源任务（即训练任务）中收集的知识自动地构建先验知识，并使用它将学习过程偏向任何新任务（即测试任务）。结果是样本数量的显著减少和所学习的解决方案的准确性显著改进。

本章的目的。与监督学习不同，强化学习问题以大量的元素为特征，比如动态和回报函

数，并且可以根据任务间的差异和相似性来定义许多不同的迁移设置。通过最近的研究，强化学习迁移已用于设计许多不同的迁移问题。然而，由于在处理这个复杂而且具有挑战性的问题时采用了非常不同的方法和观点，通常难以很清楚地描述强化学习中的最先进的迁移。本章的目的是确定主要的迁移设置是什么，并根据从源到目标任务迁移的知识类型对算法进行分类。[Taylor and Stone, 2009] 也提供了一个关于机器学习中迁移的完全调研。虽然该调研提供了对每个迁移算法的深入分析，本章的目标不是回顾文献中所有可用的算法，而是确认在强化学习中不同传输方法所共享的特性，并且把它们分享给大家。

本章结构。本章按照以下方式组织。在 5.2 节中，我们把迁移问题形式化，并且确定分类迁移算法的三个主要的维度，设置、迁移的知识和目标。然后，我们在三个不同的设置中回顾强化学习迁移的主要方法。在 5.3 节中，我们关注从源到目标的设置，从一个单源任务到一个单目标任务的迁移。在 5.4 节中，我们研究一组源任务和一个目标任务的更一般的设置。最后，在 5.5 节中，我们讨论当源和目标任务的状态 - 动作空间不同时，一般的源到目标的设置。在 5.6 节中，我们总结并讨论开放性问题。

表 5.1 本章中主要的符号清单

符号	意义
M	MDP
S_M	状态空间
A_M	动作空间
T_M	转换模型（动力学）
R_M	回报函数
\mathcal{M}	任务空间（任务 M 的集合）
Ω	M 上的概率概率分布
\mathcal{E}	环境（任务空间和分布）
\mathcal{H}	假设空间（例如，价值函数、策略）
$h \in \mathcal{H}$	假设（例如，一个价值函数、一个策略）
\mathcal{K}	知识空间（例如，样本和基函数）
$K \in \mathcal{K}$	知识（例如，采用的具体实现）
\mathcal{K}_s	源任务的知识空间
\mathcal{K}_t	目标任务的知识空间
$\mathcal{K}_{\text{transfer}}$	迁移算法返回的并在学习中使用过的知识空间
\mathcal{F}	定义在具体的状态 - 动作空间的函数空间
ϕ	状态 - 动作基函数
$\mathcal{A}_{\text{learn}}$	学习算法
$\mathcal{A}_{\text{transfer}}$	迁移算法
O	选项集合
$o \in O$	一个选项

5.2 强化学习迁移的框架和分类

迁移学习是一个普遍的问题，很难提供一个能够涵盖问题所有可能的观点和方法的正式定义。此外，尽管已经提出了许多不同的算法，在强化学习中，仍然缺少迁移的主要方法的明确分类。在本节中，我们首先介绍一般迁移的形式化，然后，我们通过三个主要的维度来对迁移方法进行分类。

5.2.1 迁移框架

在本节中，我们针对强化学习范式的监督学习，调整了 [Baxter, 2000; Silver, 2000] 介绍的形式。

如简介中所讨论的那样，迁移学习利用了从多个不同任务中收集的知识来提升新任务中学习的性能。我们将一个任务 M 定义为由元组 $\langle S_M, A_M, T_M, R_M \rangle$ 表示特征的马尔可夫决策过程（MDP）[Sutton and Barto, 1998]，其中 S_M 是状态空间， A_M 是动作空间， T_M 是转换函数， R_M 是回报函数。当状态 - 动作空间 $S_M \times A_M$ 定义了任务的域， T_M 和 R_M 定义了任务的目标。

在迁移学习问题中涉及的任务空间由 $\mathcal{M} = \{M\}$ 表示, Ω 表示任务空间 \mathcal{M} 上的概率分布, 然后, 用 $\mathcal{E} = \langle \mathcal{M}, \Omega \rangle$ 表示环境, 其定义了迁移任务中的设置。从任务分布中提取给学习器呈现的任务 (即 $M \sim \Omega$)。这个一般性的定义类似于传统监督学习的设置, 其中训练样本是从一个给定的分布中得到。因此, 类似于分类和回归, 迁移学习是基于以下思想: 由于任务是从相同的分布中得到的, 所以能够在有限数量的源任务 (或训练任务) 中基于平均得到良好性能的算法, 然后它将广泛运用于来自相同分布 Ω (或测试任务) 的目标任务 \mathcal{M} 中。

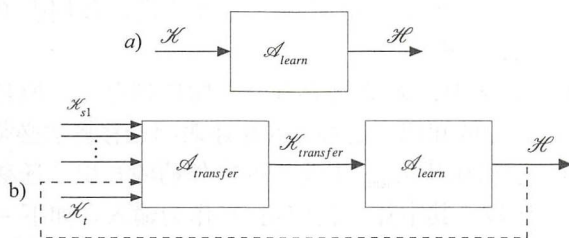


图 5.1 a) 在标准的学习过程中, 学习算法获得关于任务的知识的形式作为输入 (例如, 样本、方案结构、参数), 并且返回一个解决方案。b) 在迁移过程中, 迁移阶段首先把从一组源任务中保留的知识作为输入, 然后返回一个新的知识作为学习算法的输入。虚线表示定义一个连续过程的可能性, 从解决任务获得的经验在解决新问题中得到重用

一个标准的学习算法手动地将某种形式的任务的知识作为输入, 并且从一组可能的结果中返回一个解决方案。我们使用 \mathcal{K} 表示用作学习算法输入的知识空间, 并且 \mathcal{H} 表示可以返回的假设的空间。特别地, \mathcal{H} 指的是算法的所有元素, 其用来计算一个任务的解决方案, 特别是实例 (比如样本), 问题的表示 (比如, 选项集、特征集), 以及算法使用的参数 (比如学习率)。值得注意的是, \mathcal{H} 包括专家提供的先验知识, 从一个迁移算法中得到的迁移知识, 以及从任务中收集的直接知识。一个一般性的学习算法可以定义为映射:

$$\mathcal{A}_{\text{learn}}: \mathcal{K} \rightarrow \mathcal{H} \quad (5.1)$$

示例 1 让我们考虑带有线性函数近似的拟合 Q 迭代 [Ernst et al, 2005]。拟合 Q 迭代首先收集 N 个样本 (实例), 然后, 通过一个迭代过程返回逼近任务的最优动作 - 价值函数的一个动作 - 价值函数。在这种情况下, 假设空间 \mathcal{H} 是由领域专家所设计的一组 d 个特征 $\{\varphi_i: S \times A \rightarrow \mathbb{R}\}_{i=1}^d$ 的线性空间, 即 $\mathcal{H} = \{h(\cdot, \cdot) = \sum_{i=1}^d a_i \varphi_i(\cdot, \cdot)\}$ 。除了这个先验知识以外, 算法还接收一组 N 个样本 $\langle s, a, s', r \rangle$ 作为输入。因此, 拟合 Q 迭代使用的知识可以由空间 $\mathcal{K} = ((S \times A \times S \times R)^N, \mathcal{F}^d)$ 表示, 其中, 任一个实例 $K \in \mathcal{K}$, $K = (\langle s_n, a_n, r_n, s'_n \rangle_{n=1}^N, \{\varphi_i\}_{i=1}^d)$, $\varphi_i \in \mathcal{F}$ 。假定 $K \in \mathcal{K}$ 作为输入, 算法返回一个动作 - 价值函数 $h \in \mathcal{H}$ (即, $\mathcal{A}_{\text{FQI}}(K) = h$)。

基于之前的定义, 我们现在定义迁移学习算法的一般形式。通常, 在单任务学习中, 仅直接收集实例, 而问题的表示和参数是由专家作为先验知识给出。在迁移学习中, 其目标是减少实例的需求, 通过目标任务和来自领域专家的先验知识来调整 and 适应学习算法的结构 (即知识作为输入)。 $\mathcal{E} = \langle \mathcal{M}, \Omega \rangle$ 表示环境, L 表示从任务空间 \mathcal{M} 得到的源任务的数目, \mathcal{M} 其根据 Ω 分布得到, 一个迁移学习算法通常是知识迁移和学习阶段的结果。 \mathcal{K}_s^L 表示从源任务 L 收集到的知识和目标任务的可用知识, \mathcal{K}_t 表示目标任务已有知识, 迁移阶段定义为:

$$\mathcal{A}_{\text{transfer}}: \mathcal{K}_s^L \times \mathcal{K}_t \rightarrow \mathcal{K}_{\text{transfer}} \quad (5.2)$$

其中, $\mathcal{K}_{\text{transfer}}$ 是学习阶段被迁移的最终的知识。特别地, 学习算法现在定义为:

$$\mathcal{A}_{\text{learn}}: \mathcal{K}_{\text{transfer}} \times \mathcal{K}_i \rightarrow \mathcal{H} \quad (5.3)$$

示例 2 让我们考虑由 [Lazaric, 2008] 提出的迁移算法, 其中, 从一组 L 个源任务中学习得到一组特征。在这种情况下, $\mathcal{A}_{\text{transfer}}$ 是针对 L 个任务的每一个任务, 将 N_s 个样本作为输入, 从 \mathcal{F} 返回 d 个特征 $\{\varphi_{ij}\}_{i=1}^d$ 。然后, 使用拟合 Q 迭代算法来学习一个目标任务和 $\mathcal{A}_{\text{learn}}$ 的解决方案, 其中, $\mathcal{A}_{\text{learn}}$ 使用 N_t 个目标样本作为输入, 特征从迁移空间中获得, 最后通过特征 $\{\varphi_{ij}\}_{i=1}^d$ 返回一个在空间 \mathcal{H} 中的函数。也就是说, 我们拥有 $\mathcal{K}_s = (S \times A \times S \times R)^{N_s}$, $\mathcal{K}_t = (S \times A \times S \times R)^{N_t}$ 和 $\mathcal{K}_{\text{transfer}} = \mathcal{F}^d$ 。

虽然在公式 (5.2) 的定义中, \mathcal{K}_i 在迁移和学习阶段都存在, 但是在大多数的迁移设置中, 在迁移阶段没有关于目标的知识。这种形式化还表明迁移算法必须与第二阶段的特征的学习算法兼容, 既然 $\mathcal{K}_{\text{transfer}}$ 用作从 $\mathcal{A}_{\text{learn}}$ 中获得的额外的源知识。迁移算法的性能常常和公式 5.1 中的学习算法进行比较, 其中算法仅使用 \mathcal{K}_i 作为输入。如下一节所讨论的, 特定的设置 \mathcal{E} 、知识空间 \mathcal{H} 和性能的度量方式定义了迁移问题和方法的主要类别。

5.2.2 分类

在本节中, 我们提出强化学习中迁移的主要方法的分类。我们定义三个主要的维度: 设置、迁移的知识和目标。

5.2.2.1 设置

在迁移问题的一般化形式中, 我们定义环境 \mathcal{E} 作为任务空间 \mathcal{M} 和在其上的概率分布 Ω 。与其他的学习范式 (参见 [Pan and Yang, 2010] 对监督学习中的设置的回顾) 不同, 强化学习问题由不同的元素定义, 比如动态和回报, 根据这些元素中每一个的相似和不同而定义。比如, 在 [Mehta et al, 2008] 考虑的迁移问题中, 所有任务共享相同的状态-动作空间和动态, 但是回报函数是从作为基本回报函数和权重向量的线性组合中获得。在这种情况下, 任务空间 \mathcal{M} 是一组 MDP, 其由改变回报函数的权值来得到。此外, 尽管在一般定义中, 任务是从分布 Ω 中抽取的, 但是存在许多迁移设置, 其中任务已预先设定并且不考虑对其他任务的泛化。比如, 大多数任务间映射方法 (例如参见, [Taylor et al, 2007a]) 集中于其中仅一个源任务和一个目标任务的设置。虽然通常做出相似性的隐含假设, 但是任务可简单地定为算法的输入, 并且没有定义显示分布。在下文中, 我们将区分迁移设置的三种不同类型 (见图 5.2)。

1) 固定域中从源任务到目标任务的迁移。如 5.2.1 节所定义, 任务的域由其状态-动作空间 $S_M \times A_M$ 所确定, 而任务的特定结构和目标由动态 T_M 和回报 R_M 确定。大多数早期文献中, 强化学习中的迁移集中在设置, 其中域是固定的并且仅涉及两个任务: 源任务和目标任务。这个设置通常称为监督学习文献中的归纳迁移学习 [Pan and Yang, 2010]。迁移算法在迁移时可能会或可能不会访问目标任务。如果没有可用的目标知识, 一些迁移算法从源任务 (比如, 策略) 中收集知识从而执行浅层的迁移, 并且在目标任务中直接使用。其他算法试图从源任务中吸取一些一般性的特征 (比如, 子目标), 这些特征在求解共享相同特征的目标任务中可能相关。另一方面, 当一些目标知识在迁移时可用, 则其用于使源任务适应目标任务。比如, 在 [Taylor et al, 2008b] 中, 目标样本用于确定源和目标状态-动作变量之间的最佳映射。并且因此将源策略转换成用于目标任务中初始化学习过程的目标策略。

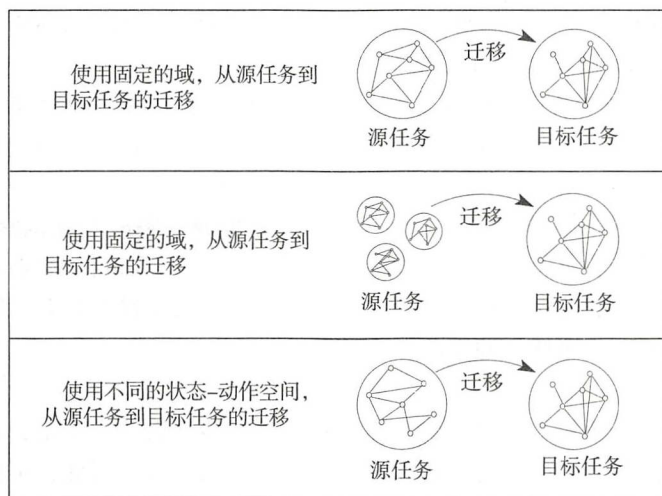


图 5.2 根据源任务数量和它们（与目标任务有关）的差异定义三个主要的迁移设置

2) 固定域中跨多任务的迁移。在此设置中，考虑在任务空间上具有分布的环境 \mathcal{E} 的一般定义。在这种情况下，任务共享相同的域，并且迁移算法把一组源任务中收集的知识作为输入，在目标任务中用其提高性能。在这个设置中，目标通常是根据分布 Ω 在 \mathcal{M} 中泛化任务。与监督学习类似，我们预计，随着源任务数量的增加，与不使用任何迁移知识的单任务学习算法相比，迁移算法能够提高目标任务的平均性能。

3) 不同域间跨任务的迁移。最后，在这个设置中，任务具有不同的域，即它们在数量和范围上可能具有不同的状态-动作变量。在这个情况下，大多数迁移方法考虑源-目标情景，并且重点关注如何定义源状态-动作变量和目标变量之间的映射，以便获得一个有效的知识迁移。

5.2.2.2 知识

迁移知识的定义和具体的迁移过程是表示一个迁移学习算法的主要方面。在 5.2.1 节的定义中，空间 \mathcal{K} 包含从环境中收集的实例（比如，样本轨迹）、解决方案的表示和算法本身的参数。一旦算法考虑的知识空间已定，重要的是设计如何实际使用该知识将信息从源任务迁移到目标任务中。[Silver, 2000; Pan and Yang, 2010] 提出了在监督学习任务中保留和迁移知识的一般分类。[Taylor and Stone, 2009] 在强化学习中引入了一个详细的迁移分类。在这里，沿着 [Lazaric, 2008] 的思路，我们提出一个更广泛的分类以识别宏观类别的方法。我们将可能的知识迁移方法分为三类：实例迁移，表示迁移，参数迁移。

1) 实例迁移。与动态规划算法（其中动态和回报函数是已知的）不同，所有的强化学习算法依赖于与 MDP 直接交互中收集的一组样本，以构建手头任务的解决方案。这组样本可用于在基于模型方法中估计 MDP 的模型，或者用于在无模型方法中直接建立一个价值函数或策略的逼近。最简单的迁移算法收集不同源任务的样本，并且在学习目标任务时重用它们。比如，轨迹样本的迁移可用来简化新任务模型的估计 [Sunmola and Wyatt, 2006] 或者动作价值函数的估计，如 [Lazaric et al, 2008]。

2) 表示迁移。每一个强化学习算法使用任务和解决方案的一个特定表示，比如状态-聚合、神经网络，或用于近似最优价值函数的一组基函数。在不同任务上学习之后，迁移算法通常执行一个抽象的过程，它改变任务和解决方案的表示。在这一类别中，许多方法采用了

不同的回报形成 [Konidaris and Barto, 2006] 和通过选择 [Singh et al, 2004] 将 MDP 扩展到基函数提取 [Mahadevan and Maggioni, 2007]。

3) 参数迁移。大多数的强化学习算法通过定义算法本身的初始化和动作的多个参数进行描述。比如, 在 Q 学习中 [Watkins and Dayan, 1992], Q 表以任意值初始化 (比如, 动作值的最高可能值, $R_{\max}/(1-\gamma)$), 并且以学习率 α 的一个梯度下降值来更新。初始值和学习率定义了算法所使用的输入参数集合。一些迁移算法根据源任务改变和调整算法参数。比如, 如果一些状态-动作对的动作值在所有源任务中十分类似, 则可以将目标任务的 Q 表初始化为更方便的值, 从而加速学习过程。尤其, 通常采用初始的解决方案 (即策略或价值函数) 的迁移来完成仅有一个单源任务的学习算法的初始化。

5.2.2.3 目标

在监督学习中, 分类器或回归器的性能通常是根据预测误差来测量, 在强化学习中, 许多可能的测量可以用于评估由学习算法返回的解决方案的好坏。因此, 可以根据多个不同的性能测量值来评估迁移算法。为了度量不同的性能, 可以使用不同的迁移指标。在 [Taylor and Stone, 2009] 中, 作者提出了许多指标来衡量单任务方法的迁移。在这里, 我们将讨论由 [Langley, 2006] (见图 5.3) 建议的一般性迁移问题的三个主要迁移目标:

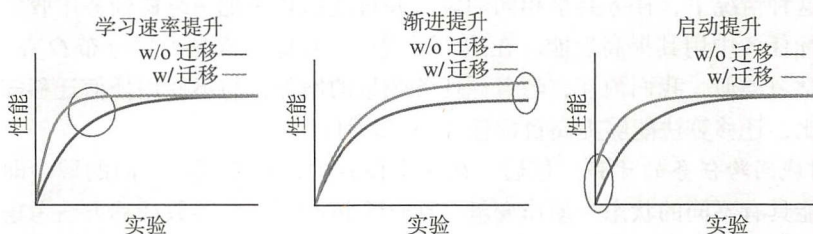


图 5.3 迁移学习 [Langley, 2006] 的三个主要目标。圆圈标识了在学习过程中, 通过使用迁移方案 (单任务方法) 获得的期望的性能提升

1) 学习速度改进。这个目标是减小学习所需的样本的数量。当根据 Ω 对新任务进行采样时, 从前一组解决了的任务中保留的知识可以用来将学习算法偏向一组有限的解决方案, 以便减少其学习时间。学习算法的复杂度通常由实现期望性能所需的样本数量来决定。在强化学习中, 该目标是根据两种不同方法来进行的。第一种方法是使算法在使用从环境中收集样本方面更加高效。比如, [Kalmar and Szepesvari, 1999; Hauskrecht, 1998] 表明, 选项的使用可以通过更新价值函数估计与选项收集的总奖励来提高值迭代备份的有效性, 从而减少收敛到一个次优解的迭代次数。第二种方法是关于收集样本的策略。在线强化学习算法中, 通过探索策略从与环境的直接交互中收集样本。通过求解一组任务收集的经验可以为新的相关任务定义更好的探索策略。比如, 如果所有的任务在状态空间的有限区域中具有目标, 则频繁访问那个区域的探索策略将带来更多的有效样本。

在实际应用中, 可以使用至少三种不同的方法来度量学习速度的提升: 时间阈值、面积比和有限样本分析。在考虑目标性能的所有问题中 (比如, 在导航问题中足够少的步骤数), 可以设置阈值并且测量多少经验 (比如, 样本、片段、迭代) 是单任务和迁移算法所需要的。如果迁移算法成功地利用从先前任务中收集的知识, 我们期望它需要更少的经验来达到目标性能。这种指标的主要缺点是阈值可以是任意的, 并且它不考虑算法的整个学习动作。事实上, 可能的情况是, 在达到一个给定的阈值时, 算法更快, 但是其具有一个非常差的初始性

能或者不能实现渐近的最优性能。由 [Taylor and Stone, 2009] 引入的面积比指标通过考虑学习曲线下的整个面积来处理这个问题。面积比定义为：

$$r = \frac{\text{迁移面积} - \text{未迁移面积}}{\text{未迁移面积}} \quad (5.4)$$

虽然这个指标成功地考虑了算法的动作，但它是与规模成比例的。比如，当每一片段的回报用作性能的一个指标时，回报的规模将影响面积比，并且回报的变化可能导致不同算法比较中的不同结论。虽然两个先前的度量可以实证比较性能，但是，通过推导算法的基于样本的边界来进行更加严格的比较也是有意义的。在这种情况下，可以根据任务的参数计算由算法返回的方法的误差的上限，并且样本的数量是已知的。比如，如果一个算法返回一个函数 $h \in \mathcal{H}$ ， Q^* 是最优动作-价值函数，一个有限的样本边界通常定义为：

$$\|h - Q^*\|_\rho \leq \varepsilon_1(\mathcal{H}, Q^*) + \varepsilon_2(N) \quad (5.5)$$

其中， ρ 是状态空间 S 上的一个分布， $\varepsilon_1(\mathcal{H}, Q^*)$ 是近似误差，它考虑了在 \mathcal{H} 中最可能方法的近似误差， $\varepsilon_2(N)$ 是估计误差，它随着样本的数量上升而降低。迁移算法应该能够减少估计误差，在与单任务算法具有相同数量的样本时，它们可以实现更好的性能。虽然强化学习许多流行的算法提供了有限样本分析，比如拟合值迭代 [Munos and Szepesvári, 2008]、LSTD [Farahmand et al, 2008; Lazaric et al, 2010] 和贝尔曼残差最小化 [Antos et al, 2008; Maillard et al, 2010]，据我们所知，目前对于强化学习中的任何迁移算法都没有有限样本分析。

正如接下来的章节中所讨论的，这个目标通常通过实例迁移来实现，即通过将源样本添加到目标任务的样本集中，以及将学习过程通过参数传递方法初始化为一个方便的解决方案。表示迁移算法通过扩展任务（比如，向动作集添加选项）和解（即，添加特征）的当前表示来实现学习速度的提升。

2) 渐近改进。在大多数实际问题中，最优价值函数或策略的完美逼近是不可能的（比如，连续状态-动作空间问题），因此，必须使用函数逼近技术。逼近越准确，收敛时的泛化（和性能）越好。逼近的精度严格依赖于用于表示解决方案（比如，价值函数）的假设 \mathcal{H} 的空间结构。这个目标修改 \mathcal{H} 的结构（比如，通过改变线性近似空间中的特征）表示迁移算法，以便精确地逼近 \mathcal{M} 中任务的解决方案。 \mathcal{H} 的质量的经验测量是比较迁移和单任务学习的渐近性能（即，当大量样本可用时）。同样在这种情况下，通过提供对其性能有限的样本分析来分析迁移算法的有效性。特别地，渐近改进相当于在公式 (5.5) 的范围内更好地逼近误差项。与学习速度改进类似，目前没有迁移算法能够保证改进平均逼近误差。

3) 快速启动改进。学习过程通常从假设空间 \mathcal{H} 中的随机值或任意假设 h 开始的。根据环境的定义，所有的任务都从分布 Ω 中得到。因此，在观察多个源任务之后，迁移算法可能在 \mathcal{M} 中的任务解决方案上构建有效的先验，并且将学习算法初始化为具有更好性能的初始化假设（与随机初始化相比）。值得注意的是，这个目标不必对应于学习速度的提升。让我们考虑其最优策略与目标任务的最优策略明显不同的源任务，但是同时，其仅实现稍微次优的性能（比如，在不同的状态空间区域中，有不同最终正回报的两个目标状态）。在这种情况下，可以通过将学习算法初始化为源任务的最优策略来获得初始性能的提升，但是这可能导致学习速度变差。事实上，初始策略不提供手头任务的实际最优策略的样本，因此，减慢了学习算法的速度。另一方面，从源任务迁移的策略可能是用于学习目标最优策略的一种有效的探索策略，但是可能得到非常差的性能。这个目标通常由参数迁移算法来执行，其中

[152]

[153]

以合适的解来初始化学习算法，它的性能与随机（任意）初始化相比更好。

5.2.2.4 调查

根据前面章节中介绍的框架，以下调查按照表 5.2 中的维度进行组织。在下面的章节中，我们首先根据所考虑的特定设置对强化学习中的主要迁移方法进行分类。在每个设置中，我们进一步根据算法从源迁移到目标的知识类型进行分类，最后，我们讨论如何实现那些目标。请注意，关于强化学习迁移的文献在三个设置上不是平均分布的。大多数关于强化学习迁移的早期文献集中于源到目标的设置，而最近研究的主流是从一组源任务迁移。最后，对映射不同状态和动作空间的问题的研究主要依赖于手动编码转换，并且有待进一步研究。

表 5.2 RL 中迁移学习的三个维度。每个迁移方法都是专门为环境设计的，它传输某种类型的知识，并追求目标。该调查根据第一维对现有算法进行分类，然后根据迁移的知识审查方法，并讨论它们实现的目标

设置	知识	目标
使用固定域从源到目标的迁移	实例	学习速率
使用固定域迁移交叉任务	表示	渐进性能
使用不同的域迁移交叉任务	参数	启动

154

5.3 固定状态 – 动作空间中从源到目标迁移的方法

在本节中，我们考虑最简单的设置，其中，迁移发生在从一个源任务到一个目标任务。我们首先在 5.3.1 节中阐述一般设置，然后，通过根据迁移知识的类型对主要方法进行分类。下面讨论的大多数方法要么改变问题的表示，要么从源到目标任务直接迁移。此外，不同于 5.4 节和 5.5 节中考虑的其他两个设置，并不是所有可能的知识迁移模型都被考虑，并且据我们所知，还不存在为这个特定设置提出的实例迁移方法。

5.3.1 问题形式化

我们定义两个 MDP，一个源任务 $M_s = \langle S, A, T_s, R_s \rangle$ 和一个目标任务 $M_t = \langle S, A, T_t, R_t \rangle$ ，共享相同的状态 – 动作空间 $S \times A$ 。环境 \mathcal{E} 由任务空间 $\mathcal{M} = \{M_s, M_t\}$ 和任务分布 Ω 所定义， M_s 是第一个任务， M_t 是第二个任务。

示例 3 让我们来考虑图 5.4 中描述的迁移问题。源任务是一个导航问题，其中，学习器应该从标记为 S 的区域移动到目标区域 G 。目标任务和源任务共享完全相同的状态 - 动作空间和相同的动态，但是初始状态和目标（以及回报函数）是不同的。迁移算法首先从与源任务交互中收集某种形式的知识，然后生成可迁移的知识，其可以用作目标任务中学习算法的输入。在该示例中，迁移算法可以利用动态中的相似性并且识别可以在学习目标任务中有用的规则。如下一节所描述的，在这种情况下执行迁移的一种有效的方式是发现在具有这样动态的环境中有助于导航的策略（即选项）。比如，图 5.4 中描述的策略驱使学习器从左侧房间的任何一点移动到两个房间之间门的位置。这样的策略有助于解决任何需要学习器从左侧房间的开始区域移动到右侧房间的目标区域的导航任务。另一种流行的方式是发现非常适合于近似环境中最优价值函数的特征。事实上，这个动态表现出对称性和不连续性，可能被保留在价值函数中。比如，源和目标的值函数在分离两个房间的墙壁附近是不连续的。因

此，一旦源任务被解决，迁移算法应分析动态和价值函数，并返回一组捕获这种不连续性并保持问题的对称性的特征。

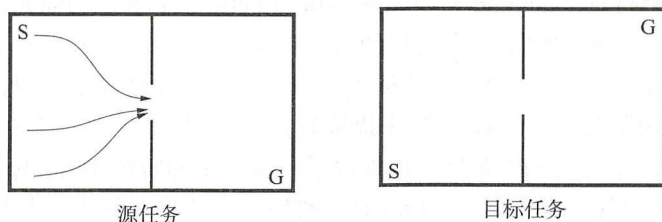


图 5.4 对从源到一个固定的状态-动作空间目标的迁移的设置实例

5.3.2 表示迁移

在一些迁移问题中，在迁移实际发生之前没有提供关于目标任务的已知知识，并且公式 (5.2) 中的 \mathcal{K}_t 总是空的。在这种情况下，重要的是从源任务抽象可能用于目标任务的一般特性。迁移算法首先从源任务中收集一些知识，然后改变空间 \mathcal{H} 或 MDP 的表示来加速目标任务中的学习。

选项发现。源-目标迁移问题最流行的方式之一是通过向可用动作集（关于强化学习的分级详见第9章）添加选项 [Sutton et al, 1999] 来改变 MDP 的表示。在离散的 MDP 中，选项不影响实现最优解的可能性（因为所有原始动作都可用，任何可能的策略仍然可以表示），但是如果它们到达有助于学习目标任务的状态空间的区域，则可能提升学习速度。所有的选项迁移方法都考虑离散 MDP、动作-价值函数表示的列表和仅在回报函数（即， $T_s = T_t$ ）不同的源和目标任务。这种想法是利用两个任务共享的动态结构，并忽略特定源回报函数的细节。这些方法中的大多数都共享一个共同的结构。首先从源任务中收集一组样本 $\langle s_t, a_t, r_t, s'_t \rangle$ ，并计算一个估计的 MDP \hat{M}_s 。基于估计动态的基础特性，识别一组相关的子目标，并且学习含有 d 个选项的集合。根据 5.2.1 节中的模型，源知识是 $\mathcal{K}_s = (S \times A \times S \times R)^{N_s}$ ，并且对任意一个特定实现 $K \in \mathcal{K}_s$ ，迁移算法返回 $\mathcal{A}_{\text{transfer}}(K) = (O, \mathcal{H})$ ，其中， $O = \{O_i\}_{i=1}^d$ ， $\mathcal{H} = \{h: S \times \{A \cup O\} \rightarrow \mathbb{R}\}$ 。学习算法可以在新的增强动作空间使用 Q 学习 [Sutton et al, 1999]，来学习到目标任务的解决方案。虽然这些迁移算法共享相同的结构，但关键点是如何识别子目标和从估计动态中学习选项。[McGovern and Barto, 2001] 将瓶颈状态的概念定义为一种状态，其经常被源任务的最优策略遍历，并且可以认为它是解决同一 MDP 中任务的关键。为图分割技术定义的指标在 [Menache et al, 2002; Simsek et al, 2005] 中用来识别连接不同状态空间区域的状态。[Hengst, 2003] 提出了一种基于访问状态概念自动开发 MAXQ 层次的方法。最后，[Bonarini et al, 2006] 提出一个心理学启发的方法，旨在识别可以轻松探索环境的状态。

动作空间迁移。在 [Sherstov and Stone, 2005] 中提出了一种涉及动作空间的表示-迁移的不同方法。使用随机任务扰动，从单一源任务人工生成一组任务，并且通过从 A 中去除在任何源任务中不是最优的所有动作来获得新的动作集合。在这种情况下，迁移算法返回一对 (A', \mathcal{H}) ，其中， A' 是原始动作空间 A 的子集，并且 $\mathcal{H} = \{h: S \times A' \rightarrow \mathbb{R}\}$ 。利用较小的动作集合，以学习策略的最优性为代价，显著地提升目标任务中的学习速度。事实上，根据人工生成的源任务去除的一些动作在目标任务中可能是最优的。尽管如此，如果源和目标任务是

相似的，并且扰动的任务与源任务足够不同，则该方法可能保留解决目标任务所需的大多数动作。

特征发现。在 [Mahadevan and Maggioni, 2007; Ferguson and Mahadevan, 2006; Ferrante et al, 2008] 中提出了一种类似于选项迁移的表示迁移方法。主要的区别在于，不同于选项，迁移算法提取假定空间 \mathcal{H} 定义的一组特征 $\{\varphi_i\}_{i=1}^d$ 。与选项迁移算法类似，假定任务共享相同的动态，并且使用估计的迁移模型 \hat{T}_s 来提取特征。当源知识 \mathcal{K}_s 是重新从 M_s 中收集的样本集，迁移的知识是 $\mathcal{K}_{\text{transfer}} = F^d$ 并且一旦提取了一组特定的特征 $\{\varphi_i\}_{i=1}^d (\varphi_i \in \mathcal{F})$ ，则将解决方案空间定义为 $\mathcal{H} = \{h(x, a) = \sum_{i=1}^d a_i \varphi_i(x, a)\}$ 。此外，在选项迁移中，其目标是提高学习速度，而特征迁移旨在实现目标价值函数的更好逼近（即渐近改进）。虽然选项迁移方法是专门为在线学习算法设计的，比如 Q 学习，特征迁移算法可以搭配任何强化学习算法，只需使用一个线性函数逼近的方案。[Mahadevan and Maggioni, 2007] 介绍了一种使用源 MDP 的估计图的拉普拉斯算子的光谱分析来生成原值函数（即特征）的方法。原值函数捕获当前任务（比如，对称性）动态下的主要的内在结构，因此其可能很好地逼近共享相同动态的任何任务的价值函数。[Ferguson and Mahadevan, 2006] 将这种方法推广到逼近动态略有不同的问题。最后，[Ferrante et al, 2008] 进一步将这种方法推广到更一般的设置，其中动态和回报函数在源和目标任务中可能不同。一种不同的方法是构建源图并提取原值函数，这个原值函数非常适用于具有相似动态和回报函数的逼近函数。

5.3.3 参数迁移

所有表示迁移的方法都依赖于源和目标任务足够相似的隐含假设，使得从源任务中提取的选项或特征可以有效地学习目标任务的解决方案。然而，可以定义许多不同的相似性概念。比如，当两个最优策略具有一些共同的部分（比如，它们都需要通过某些特定的状态去实现目标）时，我们期望选项-迁移方法可以良好地运行，而原值函数在价值函数保留迁移图的结构（比如，对称性）方面是有效的。根据 [Ferns et al, 2004; Phillips, 2006] 的研究，唯一明确的尝试是把源到目标的两个 MDP 之间的距离函数作为指标来度量迁移的预期性能。特别是他们分析了策略 π_s 从源迁移到目标任务的情况。然后使用 π_s 来初始化目标任务中的学习过程，并测量其性能。如果 MDP 是足够相似的，那么我们期望这种策略-迁移的方法实现一个跳跃式的改进。根据 5.2.1 节中介绍的， \mathcal{K}_s 是从源任务中收集的任意知识用来学习 π_s ，而迁移知识 $\mathcal{K}_{\text{transfer}}$ 仅包含 π_s 并且没有学习阶段实际发生。[Phillips, 2006] 沿着 [Ferns et al, 2004] 提出的测量定义了 M_s 和 M_t 之间的距离。特别的，距离 $d: S \rightarrow \mathbb{R}$ 被定义为：

$$d(s) = \max_{a \in A} (|R_s(s, a) - R_t(s, a)| + \gamma \mathcal{F}(d)(T_s(\cdot | s, a), T_t(\cdot | s, a))) \quad (5.6)$$

其中， $\mathcal{F}(d)$ 表示测量两个迁移分布 $T_s(\cdot | s, a)$ 和 $T_t(\cdot | s, a)$ 之间的不同的 Kantorovich 距离，其中状态距离 d 已知。已经证明，递归的公式 (5.6) 能够达到一个固定的点 d^* ，它可用作状态距离。当从源到目标迁移策略函数 π_s ，最优目标策略 π_t^* 的性能损失的上界由 d^* 限定，如：

$$\|V_t^{\pi_s} - V_t^{\pi_t^*}\| \leq \frac{2}{1-\gamma} \max_{s \in S} d^*(s) + \frac{1+\gamma}{1-\gamma} \|V_s^{\pi_s} - V_s^{\pi_t^*}\|$$

可以注意到，当迁移策略是源任务的最优化策略 π_s^* 时，其性能损失的上限为 d^* 的最大值，其考虑了两个任务的回报函数和迁移模型之间的差异。

如 5.5 节将讨论的，许多其他的参数 – 迁移方法已经在具有不同源和目标任务的设置中被研究出来了。

5.4 固定状态 – 动作空间中跨多任务迁移的方法

在上一节中，我们只有一个源任务是已知的，这里我们回顾当一组源任务是已知的时候的迁移方法的一般性设置。在这个设置中，迁移算法应处理两个主要的问题：如何合并来自不同源的知识，以及如何避免从与目标任务太多不同的源的迁移（负迁移）。

5.4.1 问题形式化

在本节中，我们考虑更一般的设置，其中环境 \mathcal{E} 用一组任务 \mathcal{M} 和一个分布 Ω 定义。与 5.3.1 节中的设置类似，这里所有的任务共享相同的状态 – 动作空间，即 $M \in \mathcal{M}$, $S_M = S$ 并且 $A_M = A$ 。虽然并非所有的方法在下一节中都明确定义了分布 Ω ，但是它们都依赖于隐含的假设，即迁移问题中涉及的所有任务在动态和回报函数中共享某些特征，通过观察一些源任务，迁移算法可以在 \mathcal{M} 的所有任务中良好地推广。

示例 4 让我们考虑 [Mehta et al, 2008] 提出的实时战略（RTS）游戏。在 RTS 中，有许多基本任务，比如攻击敌人、开采黄金、构建结构，这对完成更复杂的任务是十分有用的，比如预备军队并占领地区。更复杂的任务通常视为低级任务的组合，特定的组合还与游戏的阶段、地图的特征以及许多其他参数有关。规范化问题的一个简单方式是考虑以下情况， \mathcal{M} 中的所有任务都共享相同的状态 – 动作空间和动态，但是拥有不同的回报。特别是每一个回报函数都是一组含有 d 个基回报函数的线性集合，也就是说，对于每个任务 M ，回报定义为 $R_m(\cdot) = \sum_{i=1}^d w_i r_i(\cdot)$ ，其中 w 是权值向量， $r_i(\cdot)$ 是基回报函数。每个基回报函数编码一个特定的目标（比如，打败敌人、收集黄金），而权重代表在多目标问题中它们的组合。可以假设通过设置权值向量 w 来随机生成手头的特定任务。尤其，我们可以使用来自权重 w_M 的超参数 ψ 来定义生成的分布 Ω_ψ 。比如，超参数可以是一对 $\psi = (\mu, \Sigma)$ ，并且 Ω_ψ 可以是一个多元 d 维的高斯分布 $\mathcal{M}(\mu, \Sigma)$ 。在这个情况下，迁移算法的目标是尽可能准确地估计参数 ψ ，以便在任一新的权值向量 w_M 上有可靠的前提。

159

5.4.2 实例迁移

实例迁移算法的主要思想是源样本的迁移可以改进目标任务的学习。尽管如此，如果样本从与目标任务相差过大的源进行迁移，则可能会产生负迁移。在本节中，我们回顾了 [Lazaric et al, 2008] 中提出的迁移设置的唯一实例方法，它根据源和目标任务之间的相似性选择性地迁移样本。

令 L 表示源任务的数量，[Lazaric et al, 2008] 提出的一种算法。首先，从每一个源任务 $\mathcal{K}_s = (S \times A \times S \times R)^{N_s}$ 收集 N_s 个样本，从目标任务 $\mathcal{K}_t = (S \times A \times S \times R)^{N_t}$ 中收集 N_t 个样本 ($N_t \ll N_s$)，并且迁移算法将 \mathcal{K}_t 和 \mathcal{K}_s 作为输入。不同于包含所有源样本集合作为输出返回，该方法依赖于源和目标任务之间的相似性，以选择哪些源样本应该包含在 $\mathcal{K}_{\text{transfer}}$ 中。令 $K_{s_l} \in \mathcal{K}_s$, $k_t \in \mathcal{K}_t$ 是迁移算法可用的特定源和目标样本。假设源样本的数量足够大来构建每个源模型 \hat{M}_{s_l} 的正确的基于核的估计。给定估计模型，源任务 M_{s_l} 和目标任务 M_t 之间的相似度定义为：

$$A_{s_l} = \frac{1}{N_t} \sum_{n=1}^{N_t} \mathbb{P}(< s_n, a_n, s'_n, r_n > | \hat{M}_{s_l})$$

其中, $\mathbb{P}(< s_n, a_n, s'_n, r_n > | \hat{M}_{s_l})$ 是根据 M_{s_l} 的 (估计) 模型转换 $< s_n, a_n, s'_n, r_n >$ 的概率。这种相似度指标背后的直觉是, 迁移从可能生成目标任务的源任务中收集的样本更加方便。最后, 源样本按照与目标任务的相似性成比例地迁移。该方法通过使用针对每一个源样本效用的另一个指标进行进一步的细分, 这样, 每个源任务只迁移更有可能改进目标任务中学习性能的样本。在 [Lazaric et al, 2008] 中报告的实验中, 该方法成功地鉴定哪些源与迁移样本更相关, 从而避免负迁移。

5.4.3 表示迁移

与源到目标的迁移不同, 表示迁移的目标是从源任务中推断在 \mathcal{M} 中跨任务保留的一般特性, 可以有效地利用其来改进单任务学习的平均学习性能 (根据分布 Ω 的测量)。

选项迁移。[Bernstein, 1999] 引入了一个新的选项 (称为重用选项), 将该选项添加到可用动作集合, 并构建为从 L 个源任务学习的最优策略的组合, 然后, 重新使用以加速学习目标任务。该过程是可以重复迭代的, 以便将目标任务添加到源任务, 并相应地更新重用选项。在足够数量的源任务之后, 实验显示, 重用选项能显著地加速对新任务的学习过程。选项也用于加速学习 [Perkins and Precup, 1999], 其中, 也考虑了类似 POMDP 的框架。特别是假设提前知道集合 \mathcal{M} 和当前任务标识的信念的估计。然后, 由当前置信度加权的不同任务中的平均值计算当前任务的选项的值。虽然在这两种方法中, 任务都假定是从一个共同的分布 Ω 中抽取出来的, 但它们不提供任何关于哪些选项可以保证最佳改进的分析。[Kalmar and Szepesvari, 1999] 考虑了寻找一组选项的问题, 这减少了迭代收敛所需的最大迭代次数。不同于先前的方法, 在这种情况下, 迁移算法保证返回最佳选项集。最后, [Asadi and Huber, 2007] 提出了一种使用 MAX-Q 层次架构增强发现能力的方法。

特征迁移加速学习。一种表示迁移的不同方法是确定一个函数空间, 其可能包含能够加速学习过程或在 \mathcal{M} 中正确地逼近最优价值函数。与选项 - 迁移类似, 第一个目标通常使用表格的方法在离散 MDP 的设置中实现。在这种情况下, 函数 $\mathcal{H} = \{h: S \times A \rightarrow \mathbb{R}\}$ 的空间已经保证了计算具有任意小的逼近误差的最优行为价值函数的可能性。然而, 当状态和动作的数量很大时, 学习过程将变得很慢。在状态 - 动作空间的一部分中, 用准确地近似 \mathcal{M} 中任务的最优行为 - 价值函数的特征来扩充 \mathcal{H} 的空间可以显著地加速学习过程。在这种情况下, 迁移阶段针对每个源任务, 以一个源知识 $K_{s_l} \in \mathcal{K}_{s_l}$ 作为输入, $K_{s_l} = \{< s_n, a_n, s'_n, r_n >_{n=1}^{N_s}\}$, \mathcal{H} , 虽然没有提供关于目标任务的知识。输出一组新特征 $\mathcal{H}_{\text{transfer}} = \mathcal{F}^d$, 使得 $\mathcal{A}_{\text{transfer}}(\{K_{s_l}\}_{l=1}^L) = \{\varphi_i \in \mathcal{F}, i=1, \dots, d\}$, 其中新特征使用了学习算法添加的 \mathcal{H} 。[Foster and Dayan, 2002] 提出了一种将 MDP 分解成元素片段的自动方法。特别地, 无监督方法首先用于分析迄今为止学习的最优价值函数, 并将状态空间分解成片段。每个片段 (即子任务) 可以独立求解, 并且其价值函数用作附加特征。[Drummond, 2002] 提出了一种类似的方法, 根据源任务的动态分析识别子任务和特征。在这两种情况下, 这些方法能够识别具有高度结构化的最优价值函数的迷宫问题的有用的特征。[Madden and Howley, 2004] 介绍了一种混合表示 - 参数迁移的方法。根据在源任务中学习的 Q 表, 符号学习器生成与在 Q 表学习中使用的状态特征相比更高级别抽象定义的一组决策规则。然后, 将该表示与用于初始化目标任务的 Q 表的规则一起迁移到目标任务中。

使用特征迁移改进渐近性能。虽然前面的算法考虑了离散的 MDP，其中可以精确地计算最优解，并且目标是加速学习，但是其他方法集中在连续状态-动作空间，这种情况下，函数逼近是必须的。[Walsh et al, 2006] 考虑了一个简单的状态-聚合函数逼近方案。在这种情况下，目标是找到能够准确地逼近所述任务的所有最优价值函数的状态。在 [Lazaric, 2008] 中有类似的目标，其中迁移算法找到最佳特征集去逼近源任务，然后在求解目标任务中重用它们。与 [Argyriou et al, 2008] 类似，该算法依赖于假设近似 \mathcal{M} 中任务的价值函数的特征的小子集已知（共享稀疏假设）。一组特征 $\{\varphi_i^\theta\}_{i=1}^d$ 由参数 $\theta \in \Theta$ 和相应的线性假设空间 $\mathcal{H}_\theta = \{h(x, a) = \sum_{i=1}^d a_i \varphi_i^\theta(x, a)\}$ 决定。目标是学习定义特征空间的参数 θ ，使得仅用一小组特征来逼近任务的价值函数（即，特征使得相应的最优权重 α 具有少量的非零向量）。经验评估表明，当 \mathcal{M} 中的任务的最优价值函数可以用非常少量的特征表示时，算法能够学习它们并且获得显著的收敛改进。

5.4.4 参数迁移

与表示迁移方法不同，所有参数迁移算法在任务空间 \mathcal{M} 上显式地定义分布 Ω ，并尝试估计真实分布，以便在假设空间 \mathcal{H} 上构建先验，以此来提高初始性能（快速启动改进），并减少求解 \mathcal{M} 中任何任务所需的样本数。更正式地，大多数参数迁移方法共享以下结构：令 $\mathcal{M} = \{M_\theta, \theta \in \Theta\}$ 表示参数化的任务空间，参数 θ 和 Ω_ψ 表示任务概率分布，其中， ψ 是超参数向量。主要假设所有的任务参数 θ 根据特定任务分布 Ω_ψ 独立且相同地分布。

层次贝叶斯模型。这个问题的结构通常表示为层次贝叶斯模型（HBM），如图 5.5 所示。迁移算法从每个源任务 $M_{\theta_l} (l=1, \dots, L)$ 中抽取样本 $K_{s_l} = \{\{s_n, a_n, s'_n, r_n\}_{n=1}^{N_s}\}$ 作为输入，采样来自真实分布 Ω_{ψ^*} （即 $\theta_l \stackrel{iid}{\sim} \Omega_{\psi^*}$ ）它的真实超参数未知。在 ψ 上给定一个先验，算法求解推理问题。

$$\mathbb{P}(\psi | \{K_{s_l}\}_{l=1}^L) \propto \prod_{l=1}^L \mathbb{P}(K_{s_l} | \psi) \mathbb{P}(\psi) \quad (5.7)$$

其中 $\mathbb{P}(K_{s_l} | \psi) \propto \mathbb{P}(K_{s_l} | \theta) \mathbb{P}(\theta | \psi)$ 具有最高概率的 ψ 通常被迁移并用于初始化目标任务的学习过程。值得注意的是，学习算法必须利用迁移阶段返回的特定任务分布 Ω_ψ 的知识。通常采用用于强化学习的贝叶斯算法，比如 GPTD [Engel et al, 2005]（参见第 11 章）。

公式 (5.7) 中推理的问题同时利用了对所有任务收集的知识。因此，即使每个任务的少量样本是可用的（即 N_s 很小），该算法仍然可以利用大量任务（即 L 很大）来求解推理问题并学习 ψ^* 的准确估计。随着 L 的增加，超参数 ψ 越来越接近真实的超参数 ψ^* ，并且 ψ 可以用于对从分布 Ω_{ψ^*} 中吸取的任何新目标任务在参数 θ 上构建先验。根据 θ 和 Ω_ψ 的特定定义以及求解推理问题的方式，可以从一般模型推导出许多不同的算法。

迁移推理。[Tanaka and Yamamura, 2003] 提出了一种更简单的方法。尽管假定 MDP 是来自分布 Ω ，但是所提出的算法并不试图估计任务分布，而仅仅计算关于动作值的统计。计算不同任务上的动作值的均值和方差，然后用于初始化新任务的 Q 表。[Sunmola and Wyatt,

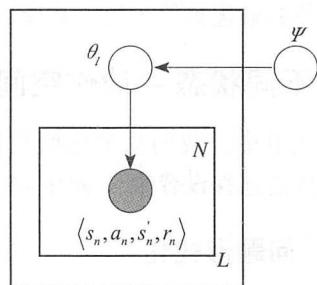


图 5.5 一个层次化贝叶斯模型（HBM）的生成模型示例。观察 $\langle s, a, s', a' \rangle$ 是根据一个由参数向量 θ 参数化的 MDP 生成的，该任务根据一个由超参数集合 ψ 定义的分生成。 ψ_0 是定义超参数先验的参数向量

162

163

2006; Wilson et al, 2007] 提出了 MDP 动态和回报函数, 它们被参数向量 θ 参数化, 并且假定它们是从共同的分布 Ω_ψ 中来。通过在超参数 ψ 上选择适当的共轭先验来求解推理问题。[Li et al, 2009] 提出 POMDP 上的迁移问题。在这种情况下, 不提供任务的显式参数化。另一方面, 它是基于历史的策略空间 \mathcal{H} , \mathcal{H} 由向量参数 $\theta \in \Theta$ 参数化。然后将 Dirichlet 过程作为非参先验用于不同任务最优策略的参数上。[Lazaric and Ghavamzadeh, 2010] 通过考虑由给定特征集合 $\mathcal{H} = \{h(x, a) = \sum_{i=1}^d \theta_i \phi_i(x, a)\}$ 构成的线性函数的空间来考虑价值函数的参数化空间的情况。假定向量 θ 来自带参数 ψ 的多元高斯分布, 而 ψ 从正反 wishart 超先验 (即 $\theta \sim \mathcal{N}(\mu, \Sigma)$) 和 $(\mu, \Sigma \sim \mathcal{N}\text{-}\mathcal{FW}(\psi))$ 中抽取。EM 算法利用了共轭先验, 使用类似 EM 的算法来求解推理问题。该方法进一步扩展到任务来自不同分布的情况。为了将任务分组到不同的类中, [Lazaric and Ghavamzadeh, 2010] 在层次贝叶斯模型的顶部放置了一个 Dirichlet 过程, 并且通过使用 Gibbs 采样求解推理问题以自动学习不同类中的数量和任务配置。最后, [Mehta et al, 2008] 将回报函数定义为回报特征的线性组合, 其在任务之间是共有的, 而权重是每个任务的特定值。权重从分布 Ω 中进行抽取, 迁移算法紧密地存储运用回报函数的源任务的最优价值函数, 并使用它们来初始化目标任务的解。

5.5 不同状态 – 动作空间中从源到目标任务迁移的方法

前几节中, 我们只考虑所有任务共享相同域 (即, 它们具有相同的状态 – 动作空间)。在最一般的迁移设置中, \mathcal{M} 中的任务可能在状态 – 动作变量的数量和范围方面不同。

5.5.1 问题形式化

164 虽然我们每一个任务 $M \in \mathcal{M}$ 定义为 $\text{MDP}\langle S_M, A_M, T_M, R_M \rangle$, 并且通过 \mathcal{M} 上的分布 Ω 来获得环境 \mathcal{E} , 但实际上只考虑了从源到目标的迁移设置。那就是说, 与 5.3 节类似, 我们考虑一个任务空间 $\mathcal{M} = \{M_s, M_t\}$, 其中源任务 $M_s = \langle S_s, A_s, T_s, R_s \rangle$ 且目标任务 $M_t = \langle S_t, A_t, T_t, R_t \rangle$, 我们想要去提升学习的性能。根据 5.2.1 节中介绍的符号, \mathcal{K}_s 和 \mathcal{K}_t 现在在不同的状态 – 空间上定义, 同时, \mathcal{K}_s 中的知识不能被学习算法直接用于目标任务上的学习。因此, 由 $\mathcal{A}_{\text{transfer}}$ 实现的迁移阶段必须返回与 \mathcal{K}_t 兼容的知识 $\mathcal{K}_{\text{transfer}}$ 。这个目标通常由以下三种不同的方式来实现: 通过手工编码映射把 \mathcal{K}_s 转换为 \mathcal{K}_t , 学习从 M_s 到 M_t 的映射并将 \mathcal{K}_s 转换为 \mathcal{K}_t , 从 \mathcal{K}_s 中提取一些可以被重用来求解 M_t 的抽象知识。在下一节中, 我们仍然遵循前面章节中使用的不同类别, 但是我们将明确使用映射问题中的某一种。

示例 5 让我们考虑由 [Taylor et al, 2008a] 介绍的在图 5.6 中的山地车域和源与目标任务。虽然问题在某种程度上是类似的 (即动态不足的车必须从谷底移动到山顶), 两个任务被定义在不同的状态空间中, 并且动作空间包含不同数量的动作。事实上, 在二维山地车任务中, 状态空间由位置和速度变量 (x, \dot{x}) 定义, 并且动作空间包含动作 $A = \{\text{Left}, \text{Neutral}, \text{Right}\}$ 。另一方面, 三维任务有两个额外的变量描述 y 中的位置以及对应的速度 \dot{y} , 动作空间变为 $A = \{\text{Neutral}, \text{West}, \text{East}, \text{South}, \text{North}\}$ 。到目前为止, 迁移算法在这里都不能使用, 因为它们从源任务迁移的知识 $\mathcal{K}_{\text{transfer}}$ 与目标任务不兼容。在这种情况下, 迁移算法必须定义源和目标的状态和动作空间之间合适的映射, 然后再用二维山地车学习迁移方案来初始化三维任务中的学习过程。

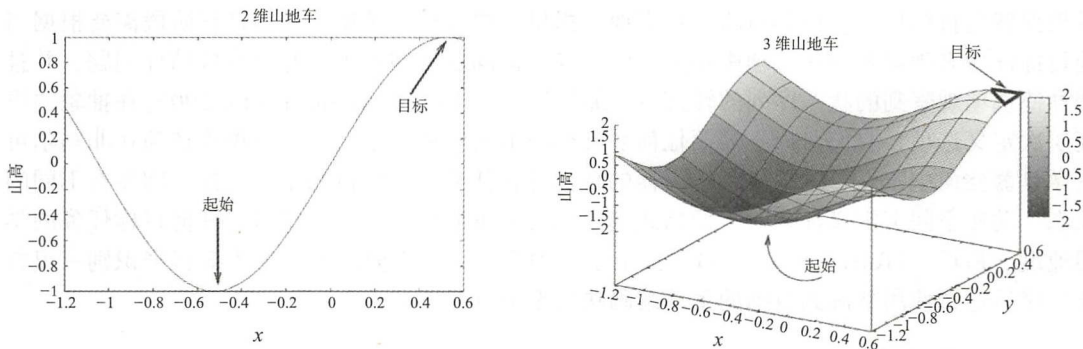


图 5.6 从二维到三维山地车的迁移 [Taylor et al, 2008a]

165

5.5.2 实例迁移

类似于 [Lazaric et al, 2008] 介绍的方法, [Taylor et al, 2008a] 研究样本的迁移。与 [Lazaric et al, 2008] 不同的是, 这里只考虑一个源任务且没有实现用于样本选择的具体特定的方法。另一方面, 这两种方法都不共享相同的域, 因此需要在它们的状态-动作变量之间显式地映射。提供手动编码映射作为迁移算法的输入, 简单地将其应用于源样本, 从而获得可用于学习目标任务的样本。由 [Taylor et al, 2007a] 引入并由 [Taylor et al, 2008a] 使用的任务间映射形式化之后, 手工编码映射由两个函数 \mathcal{X}_S 和 \mathcal{X}_A 定义。令状态空间 S_s 和 S_t 分别在 d_s 和 d_t 状态变量中因式分解 (即 $S_s = S^1 \times \cdots \times S^{d_s}$ 和 $S_t = S^1 \times \cdots \times S^{d_t}$), 并且 A_s 和 A_t 是分别具有值 $A_s = \{a^1, \dots, a^{k_s}\}$ 和 $A_t = \{a^1, \dots, a^{k_t}\}$ 的标量空间。状态映射在目标状态空间中源状态变量 $1 \leq i \leq d_s$ 的索引映射到目标状态空间变量 $1 \leq j \leq d_t$, 即 $\mathcal{X}_S: \{1, \dots, d_s\} \rightarrow \{1, \dots, d_t\}$ 。利用符号, 我们用 $\mathcal{X}_S(s) \in S_t$ 表示状态 $s \in S_s$ 到根据 \mathcal{X}_S 将每个变量 s 映射到目标变量所获得状态的转换。类似的, 动作映射 \mathcal{X}_A 将 A_s 中的每个源动作映射到 A_t 中的一个目标动作。结果, 如果 \mathcal{X}_s 是源样本空间, 并且 $\mathcal{X}_s \in \mathcal{X}_s$ 是 N_s 个样本的一个特定实现, 对于任意样本 $\langle s_n, a_n, s'_n, r_n \rangle \in K_s$, 迁移算法返回新的目标样本 $\langle \mathcal{X}_S(s_n), \mathcal{X}_A(a_n), \mathcal{X}_S(s'_n), r_n \rangle$ 。虽然 [Lazaric et al, 2008] 提出了一种算法, 其中迁移的样本在批处理的无模型强化学习算法中使用, [Taylor et al, 2008a] 研究了如何基于模型的算法拟合 R-max, 当根据源到目标任务的手工编码映射进行变换时, 可来自源任务的样本获得好处。特别是该方法在山地车问题中经证实是有效的。

5.5.3 表示迁移

如前所述, 许多迁移方法开发了可以有效地在目标任务中重用的选项。在这种情况下, 主要的问题是在源任务中学习选项从 S_s 到 A_s 的映射, 并且它们不能够在具有不同状态-动作变量的目标任务中使用。许多迁移算法通过考虑可以在不同任务中重用的抽象选项来处理这个问题。[Ravindran and Barto, 2003; Soni and Singh, 2006] 使用同态框架将任务映射到一个公共抽象层次。比如, 让我们考虑在一个空的正方形房间中的所有导航问题, 在这种情形下, 可以定义一个公共的抽象 MDP, 并且通过简单使用诸如平移、缩放和旋转的运算符来获得任何特定的 MDP。为了处理这种情形, [Ravindran and Barto, 2003] 提出了相对化选项的概念。与传统选项不同, 相对化选项是在抽象 MDP 上定义的, 没有绝对的参考框架, 然

166

后根据特定目标任务转换其策略。特别地, 提供一组可能的转换, 并且迁移阶段需要根据当前目标任务来确定相对化选项中最适合的转换。该问题可转换为贝叶斯参数估计问题, 并且选择使选项观察到的状态序列的转换更可能的转换。[Konidaris and Barto, 2007] 在抽象的更高层次定义选项, 它们可以不需要任何明确的映射或转换。事实上, 便携式选项在非马尔可夫学习器空间中定义, 这取决于学习器的特性并在任务之间保持固定。这样, 即使在不同的状态-动作空间上定义任务时, 便携式选项也可以重复使用, 以加快目前任何目标任务的学习速度。最后, [Torrey et al., 2006] 提出了一种算法, 首先使用归纳逻辑编程来识别一组技能, 然后通过使用从源到目标的手动编码映射重用目标任务。

5.5.4 参数迁移

在 5.3 节所述的设置中, 从源到目标任务的初始化解方案 (比如, 最优源策略) 的迁移是微不足道的, 在这种情况下, 使得迁移有效的关键是找到从源状态-动作空间 $S_s \times A_s$ 到目标状态-动作空间 $S_t \times A_t$ 适合的映射。

在下面回顾的大多数算法考虑手工编码映射并且研究知识 (比如, 策略、价值函数) 的不同来源的迁移如何影响目标任务的性能。通过手工编码映射和源任务的价值函数的迁移来初始化目标任务的转换由 [Taylor and Stone, 2005; Taylor et al, 2005] 首先提出, 并且已经研究了它在许多具有挑战性的问题中的影响, 比如模拟 keep-away 问题 [Stone et al, 2005]。[Banerjee and Stone, 2007] 也考虑了在一般游戏环境中价值函数的迁移, 其中不同的游戏可以通过一个共同的抽象结构来表示。[Torrey et al, 2005] 通过重复使用通过手工编码映射建议 (即, 在源任务中具有更高 Q 值的动作) 来学习目标任务中的 Q 表。虽然先前的方法假定在源和目标任务中使用相同的解决方案 (比如, 表格方法), 但是 [Taylor and Stone, 2007] 考虑将解决方案 (即价值函数或策略) 映射到另一个解决方案的问题, 当近似架构 (比如, CMAC 和神经网络) 或学习算法本身 (比如, 基于值和策略搜索方法) 在源和目标任务之间改变时。在使用与状态-动作映射类似的映射时, 迁移仍然是可能的, 并且有益于提高目标任务的性能。最后, [Taylor et al, 2007b] 研究源策略的迁移, 它使用手工编码映射把源策略转换为目标任务的有效策略, 然后使用策略搜索算法来改进它。

167

与以前的方法不同, 一些迁移算法自动识别源和目标任务之间最适合的映射。[Taylor et al, 2008] 提出了 MASTER 算法。其目标是在源到目标状态变量的所有可能的映射之间进行识别, 保证对环境动态的最佳预测。该算法从源任务接收相对大量的样本作为输入, 但只有很少的目标样本。令 X 是 S_s 和 S_t 之间所有可能映射的集合, $K_s \in \mathcal{K}_s$, $K_t \in \mathcal{K}_t$ 分别是源和目标样本的特定实现。从 K_t 中, 算法首先计算目标动态 \hat{T}_t 的近似值。然后, K_s 中的每一个样本 $\langle s_n, a_n, s'_n, r_n \rangle$ 使用可能的映射 $\chi_s \in X$ 中的一个进行转换, 然后用来计算由 $\hat{T}_t(\chi_s(s_n), a)$ 预测的状态 $\chi_s(s'_n)$ 。然后使用映射 χ_s , 其是 K_s 中样本转换预测最准确的, 用于迁移源任务的解决方案。[Talvitie and Singh, 2007] 首先学习了源任务的策略, 然后通过从源到目标状态变量的每个可能的映射来做映射策略, 从而获得目标策略。选择最适合的映射的问题随后被转换为评估目标策略中最优策略的问题。然后使用类似专家算法来解决这个问题 [Cesa-Bianchi and Lugosi, 2006]。

5.6 总结和开放性问题

在本章中, 我们为强化学习范式中的迁移学习问题定义了一个一般的框架, 提出了一个对问题的不同方法的分类, 并且回顾了文献中可用的主要方法。虽然已经提出了许多算法,

但是迁移问题远未得到解决。接下来，我们列出了几个与这个主题的研究进展相关的开放性问题。我们引用了 [Taylor and Stone, 2009] 对强化学习中其他迁移研究的调查。

迁移算法的理论分析。虽然实验结果支持强化学习算法可以从相关任务的迁移受益的想法，但是针对强化学习的迁移算法没有很强的理论保证。最近在监督学习范例中的迁移和多任务学习的研究实现了令人感兴趣的理论结果，迁移算法有望提高单个任务学习的性能。[Crammer et al, 2008] 研究了来自不同分类任务的学习重用样本的性能，并且证明了当源任务的样本分布与目标分布相比没有太大差别时，迁移方法比仅用目标样本表现更好。[Baxter, 2000] 研究了为给定的一组任务学习最适合的一组假设的问题。特别是表明了随着源任务数量的增加，迁移算法设法识别可能包含 \mathcal{M} 中所有任务的良好假设的假设集合。[Ben-David and Schuller-Borbely, 2008] 考虑了在多任务学习的上下文中学习最佳假设集的问题，其中目标不是在新任务中推广，而是在源任务中实现更好的平均性能。同时，新的理论结果可用于许多流行的强化学习算法，比如拟合值迭代 [Munos and Szepesvári, 2008]、LSTD [Farahmand et al, 2008; Lazaric et al, 2010] 和贝尔曼残差最小化 [Antos et al, 2008; Maillard et al, 2010]。一个有趣的研究方向是利用监督学习设置中的迁移算法的理论结果和单任务情况下强化学习算法的理论结果来开发新的强化学习迁移算法。

168

迁移探索学习。通过更好地使用样本（比如，通过改变假设集合），而不是通过收集更多的信息样本来实现这一目标（参见 5.2.2 节）。这个问题与探索开发困境严格相关，其目标是在不同策略的探索和迄今为止最优策略的开发之间进行权衡。最近，[Bartlett and Tewari, 2009; Jaksch et al, 2010] 研究了单任务学习的最优探索策略。尽管大多数基于选项迁移的方法隐式地偏向探索策略，但是基于先前相关任务的知识应如何对一个任务进行探索的问题是迄今很少受到关注的问题。

概念漂移和持续学习。迁移学习的一个主要假设是在 \mathcal{M} 中任务之间是可能做到明确区分的。尽管如此，在许多有趣的应用中，源和目标任务之间没有明显的区别，而是本身随着时间变化。这个问题叫作概念漂移，与持续学习和终身学习范式 [Silver and Poirier, 2007] 密切相关。其中，随着学习器自动地发现非静止环境的新区域，它能够增加解决任务的能力。虽然来自迁移学习的工具能在这种情况下重用，但需要新的方法来处理环境的非平稳性并跟踪任务的变化。

参考文献

- Antos, A., Szepesvári, C., Munos, R.: Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning Journal* 71, 89–129 (2008)
- Argyriou, A., Evgeniou, T., Pontil, M.: Convex multi-task feature learning. *Machine Learning Journal* 73(3), 243–272 (2008)
- Asadi, M., Huber, M.: Effective control knowledge transfer through learning skill and representation hierarchies. In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-2007)*, pp. 2054–2059 (2007)
- Banerjee, B., Stone, P.: General game learning using knowledge transfer. In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-2007)*, pp. 672–677 (2007)
- Bartlett, P.L., Tewari, A.: Regal: a regularization based algorithm for reinforcement learning in weakly communicating mdps. In: *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*, pp. 35–42. AUAI Press, Arlington (2009)

169

- Baxter, J.: A model of inductive bias learning. *Journal of Artificial Intelligence Research* 12, 149–198 (2000)
- Ben-David, S., Schuller-Borbely, R.: A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning Journal* 73(3), 273–287 (2008)
- Bernstein, D.S.: Reusing old policies to accelerate learning on new mdps. Tech. rep., University of Massachusetts, Amherst, MA, USA (1999)
- Bonarini, A., Lazaric, A., Restelli, M.: Incremental Skill Acquisition for Self-motivated Learning Animats. In: Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J.C.T., Marocco, D., Meyer, J.-A., Miglino, O., Parisi, D. (eds.) *SAB 2006. LNCS (LNAI)*, vol. 4095, pp. 357–368. Springer, Heidelberg (2006)
- Cesa-Bianchi, N., Lugosi, G.: *Prediction, Learning, and Games*. Cambridge University Press (2006)
- Crammer, K., Kearns, M., Wortman, J.: Learning from multiple sources. *Journal of Machine Learning Research* 9, 1757–1774 (2008)
- Drummond, C.: Accelerating reinforcement learning by composing solutions of automatically identified subtasks. *Journal of Artificial Intelligence Research* 16, 59–104 (2002)
- Engel, Y., Mannor, S., Meir, R.: Reinforcement learning with Gaussian processes. In: *Proceedings of the 22nd International Conference on Machine Learning (ICML-2005)*, pp. 201–208 (2005)
- Ernst, D., Geurts, P., Wehenkel, L.: Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research* 6, 503–556 (2005)
- Farahmand, A.M., Ghavamzadeh, M., Szepesvári, C., Mannor, S.: Regularized policy iteration. In: *Proceedings of the Twenty-Second Annual Conference on Advances in Neural Information Processing Systems (NIPS-2008)*, pp. 441–448 (2008)
- Fawcett, T., Callan, J., Matheus, C., Michalski, R., Pazzani, M., Rendell, L., Sutton, R. (eds.): *Constructive Induction Workshop at the Eleventh International Conference on Machine Learning* (1994)
- Ferguson, K., Mahadevan, S.: Proto-transfer learning in markov decision processes using spectral methods. In: *Workshop on Structural Knowledge Transfer for Machine Learning at the Twenty-Third International Conference on Machine Learning* (2006)
- Ferns, N., Panangaden, P., Precup, D.: Metrics for finite markov decision processes. In: *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI-2004)*, pp. 162–169 (2004)
- Ferrante, E., Lazaric, A., Restelli, M.: Transfer of task representation in reinforcement learning using policy-based proto-value functions. In: *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2008)*, pp. 1329–1332 (2008)
- Foster, D.J., Dayan, P.: Structure in the space of value functions. *Machine Learning Journal* 49(2-3), 325–346 (2002)
- Gentner, D., Loewenstein, J., Thompson, L.: Learning and transfer: A general role for analogical encoding. *Journal of Educational Psychology* 95(2), 393–408 (2003)
- Gick, M.L., Holyoak, K.J.: Schema induction and analogical transfer. *Cognitive Psychology* 15, 1–38 (1983)
- Hauskrecht, M.: Planning with macro-actions: Effect of initial value function estimate on convergence rate of value iteration. Tech. rep., Department of Computer Science, University of Pittsburgh (1998)
- Hengst, B.: *Discovering hierarchy in reinforcement learning*. PhD thesis, University of New South Wales (2003)
- Jaksch, T., Ortner, R., Auer, P.: Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research* 11, 1563–1600 (2010)
- Kalmar, Z., Szepesvari, C.: An evaluation criterion for macro-learning and some results. Tech. Rep. TR-99-01, Mindmaker Ltd. (1999)
- Konidaris, G., Barto, A.: Autonomous shaping: knowledge transfer in reinforcement learning. In: *Proceedings of the Twenty-Third International Conference on Machine Learning (ICML-2006)*, pp. 489–496 (2006)

- Konidaris, G., Barto, A.G.: Building portable options: Skill transfer in reinforcement learning. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-2007), pp. 895–900 (2007)
- Langley, P.: Transfer of knowledge in cognitive systems. In: Talk, Workshop on Structural Knowledge Transfer for Machine Learning at the Twenty-Third International Conference on Machine Learning (2006)
- Lazaric, A.: Knowledge transfer in reinforcement learning. PhD thesis, Poltecnico di Milano (2008)
- Lazaric, A., Ghavamzadeh, M.: Bayesian multi-task reinforcement learning. In: Proceedings of the Twenty-Seventh International Conference on Machine Learning, ICML-2010 (2010) (submitted)
- Lazaric, A., Restelli, M., Bonarini, A.: Transfer of samples in batch reinforcement learning. In: Proceedings of the Twenty-Fifth Annual International Conference on Machine Learning (ICML-2008), pp. 544–551 (2008)
- Lazaric, A., Ghavamzadeh, M., Munos, R.: Finite-sample analysis of lstd. In: Proceedings of the Twenty-Seventh International Conference on Machine Learning, ICML-2010 (2010)
- Li, H., Liao, X., Carin, L.: Multi-task reinforcement learning in partially observable stochastic environments. *Journal of Machine Learning Research* 10, 1131–1186 (2009)
- Madden, M.G., Howley, T.: Transfer of experience between reinforcement learning environments with progressive difficulty. *Artificial Intelligence Review* 21(3-4), 375–398 (2004)
- Mahadevan, S., Maggioni, M.: Proto-value functions: A laplacian framework for learning representation and control in markov decision processes. *Journal of Machine Learning Research* 38, 2169–2231 (2007)
- Maillard, O.A., Lazaric, A., Ghavamzadeh, M., Munos, R.: Finite-sample analysis of bellman residual minimization. In: Proceedings of the Second Asian Conference on Machine Learning, ACML-2010 (2010)
- McGovern, A., Barto, A.G.: Automatic discovery of subgoals in reinforcement learning using diverse density. In: Proceedings of the Eighteenth International Conference on Machine Learning, ICML 2001 (2001)
- Mehta, N., Natarajan, S., Tadepalli, P., Fern, A.: Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning Journal* 73(3), 289–312 (2008)
- Menache, I., Mannor, S., Shimkin, N.: Q-cut - dynamic discovery of sub-goals in reinforcement learning. In: Proceedings of the Thirteenth European Conference on Machine Learning, pp. 295–306 (2002)
- Munos, R., Szepesvári, C.: Finite time bounds for fitted value iteration. *Journal of Machine Learning Research* 9, 815–857 (2008)
- Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22(22), 1345–1359 (2010)
- Perkins, D.N., Salomon, G., Press, P.: Transfer of learning. In: *International Encyclopedia of Education*. Pergamon Press (1992)
- Perkins, T.J., Precup, D.: Using options for knowledge transfer in reinforcement learning. Tech. rep., University of Massachusetts, Amherst, MA, USA (1999)
- Phillips, C.: Knowledge transfer in markov decision processes. McGill School of Computer Science (2006), <http://www.cs.mcgill.ca/~martin/usrs/phillips.pdf>
- Ravindran, B., Barto, A.G.: Relativized options: Choosing the right transformation. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003), pp. 608–615 (2003)
- Sherstov, A.A., Stone, P.: Improving action selection in MDP's via knowledge transfer. In: Proceedings of the Twentieth National Conference on Artificial Intelligence, AAAI-2005 (2005)
- Silver, D.: Selective transfer of neural network task knowledge. PhD thesis, University of Western Ontario (2000)
- Silver, D.L., Poirier, R.: Requirements for Machine Lifelong Learning. In: Mira, J., Álvarez, J.R. (eds.) *IWINAC 2007, Part I. LNCS*, vol. 4527, pp. 313–319. Springer, Heidelberg (2007)

- Simsek, O., Wolfe, A.P., Barto, A.G.: Identifying useful subgoals in reinforcement learning by local graph partitioning. In: Proceedings of the Twenty-Second International Conference of Machine Learning, ICML 2005 (2005)
- Singh, S., Barto, A., Chentanez, N.: Intrinsically motivated reinforcement learning. In: Proceedings of the Eighteenth Annual Conference on Neural Information Processing Systems, NIPS-2004 (2004)
- Soni, V., Singh, S.P.: Using homomorphisms to transfer options across continuous reinforcement learning domains. In: Proceedings of the Twenty-first National Conference on Artificial Intelligence, AAAI-2006 (2006)
- Stone, P., Sutton, R.S., Kuhlmann, G.: Reinforcement learning for RoboCup-soccer keep-away. *Adaptive Behavior* 13(3), 165–188 (2005)
- Sunmola, F.T., Wyatt, J.L.: Model transfer for markov decision tasks via parameter matching. In: Proceedings of the 25th Workshop of the UK Planning and Scheduling Special Interest Group, PlanSIG 2006 (2006)
- Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
- Sutton, R.S., Precup, D., Singh, S.: Between mdps and semi-mdps: a framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112, 181–211 (1999)
- Talvitie, E., Singh, S.: An experts algorithm for transfer learning. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-2007), pp. 1065–1070 (2007)
- Tanaka, F., Yamamura, M.: Multitask reinforcement learning on the distribution of mdps. In: IEEE International Symposium on Computational Intelligence in Robotics and Automation, vol. 3, pp. 1108–1113 (2003)
- Taylor, M.E., Stone, P.: Behavior transfer for value-function-based reinforcement learning. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2005), pp. 53–59 (2005)
- Taylor, M.E., Stone, P.: Representation transfer for reinforcement learning. In: AAAI 2007 Fall Symposium on Computational Approaches to Representation Change during Learning and Development (2007)
- Taylor, M.E., Stone, P.: Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research* 10(1), 1633–1685 (2009)
- Taylor, M.E., Stone, P., Liu, Y.: Value functions for RL-based behavior transfer: A comparative study. In: Proceedings of the Twentieth National Conference on Artificial Intelligence, AAAI-2005 (2005)
- Taylor, M.E., Stone, P., Liu, Y.: Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research* 8, 2125–2167 (2007a)
- Taylor, M.E., Whiteson, S., Stone, P.: Transfer via inter-task mappings in policy search reinforcement learning. In: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS-2007 (2007b)
- Taylor, M.E., Jong, N.K., Stone, P.: Transferring instances for model-based reinforcement learning. In: Proceedings of the European Conference on Machine Learning (ECML-2008), pp. 488–505 (2008a)
- Taylor, M.E., Kuhlmann, G., Stone, P.: Autonomous transfer for reinforcement learning. In: Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2008), pp. 283–290 (2008b)
- Thorndike, E.L., Woodworth, R.S.: The influence of improvement in one mental function upon the efficiency of other functions. *Psychological Review* 8 (1901)
- Torrey, L., Walker, T., Shavlik, J., Maclin, R.: Using Advice to Transfer Knowledge Acquired in one Reinforcement Learning Task to Another. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 412–424. Springer, Heidelberg (2005)
- Torrey, L., Shavlik, J., Walker, T., Maclin, R.: Skill Acquisition Via Transfer Learning and Advice Taking. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 425–436. Springer, Heidelberg (2006)

- Utgoff, P.: Shift of bias for inductive concept learning. *Machine Learning* 2, 163–190 (1986)
- Walsh, T.J., Li, L., Littman, M.L.: Transferring state abstractions between mdps. In: *ICML Workshop on Structural Knowledge Transfer for Machine Learning* (2006)
- Watkins, C., Dayan, P.: Q-learning. *Machine Learning* 8, 279–292 (1992)
- Wilson, A., Fern, A., Ray, S., Tadepalli, P.: Multi-task reinforcement learning: a hierarchical bayesian approach. In: *Proceedings of the Twenty-Forth International Conference on Machine learning (ICML-2007)*, pp. 1015–1022 (2007)

探索的样本复杂度边界

Lihong Li

摘要

众所周知，高效探索是强化学习中一个根本性的挑战。能高效地对策略空间进行探索的算法一般能更快地收敛到近似最优策略。然而，尽管启发式技术在实践中用得普遍，但是这些技术缺乏正式的理论支撑，所以并不能保证总是有效的。本章将采用一致的方法来研究具有多项式样本探索复杂度的算法，其中将包含基于模型的算法以及模型无关的算法。这些算法简称为 PAC-MDP 算法，它们在大部分情况下都有近似最优的表现。一个名为 KWIK 的新的学习模型通常用于统一大多数现有的基于模型的 PAC-MDP 算法，而这些算法则用于对马尔可夫决策过程的各种子类的划分。我们还将样本复杂性框架与替代方案进行比较，用于形式化探索效率，例如遗憾最小化和贝叶斯最优解。

6.1 简介

广义上，在线强化学习 (RL) 的学习器通过试错的方式和初始未知环境进行交互以求最大化可累积回报。如果没有外部监督，学习器需要在所处环境中进行试验以便获知基于怎样的原则进行决策才能够实现自身目标的最优化。一个学习器要想最大化回报，就必须有目的性的去尝试各种不同的动作，即便有时候这些动作看起来不算最优，但这样做却能够收集关于其自身所处环境的信息，而这些信息将会帮助学习器在未来获得更多的回报。如果一个学习器仅仅是盲目地在所处环境中进行探索（这种学习器称为 Exploration Agent，即探索学习器）是不可行的，因为这样的一个毫无实用性的学习器通过对环境的暴力探索得到的回报将少得可怜。同样，如果一个学习器单纯地依靠自身的经验选择最优的策略（这种学习器称为 Exploiting Agent，即开发学习器）也是不可行的，因为这样的动作缺乏对所处环境的相对完整知识参考，最终很可能走向了局部最优行为策略的死胡同。因此如何平衡探索和开发这两者以实现高效学习器是强化学习中最为本质的问题 [Sutton and Barto, 1998]。

学习器对环境充分探索的必要性已经得到广泛认可认同，比如，时序差分算法要实现典型性渐近收敛就必须满足在所有状态中无限频繁地采取每个动作 [Jaakkola et al, 1994; Singh et al, 2000; Watkins and Dayan, 1992]。没有无限的探索，评估函数的初始误差将可能不能得到修正，从而导致时序差分算法收敛到某一个局部最优策略上。

现在已经有一些探索技术在实践中得到了广泛的应用 [Sutton and Barto, 1998]。但很不幸，虽然这些技术策略在某些应用中表现不错，但实际上这些技术策略并没有理论支撑所以不能保证它们总是有效的。事实上，有些探索技术甚至被证明在某些简单的问题中其性能是效率低下的，这个证明会在下一节进行展示。最近，在形式化探索效率的概念以及开发具有可证明有效的探索算法方面已经引起越来越多的关注。这种类型的分析通常集中于 RL 算

法的累积奖励究竟能有多快的速度接近最优策略。这样的保证与传统的渐近保证非常不同 [Bertsekas and Tsitsiklis, 1996], 因为它们能表征算法究竟能多快地学习在有限多个步骤之后最大化累积回报。

本章内容如下: 6.2 节对在线强化学习进行形式化, 描述一些经常用到的探索技术, 同时展示这些探索技术在某些简单的问题中是如何失效的。6.3 节讨论一些对学习器的探索效率进行量化表征的方法。由于篇幅问题, 本章将仅关注样本复杂度问题, 我们还将与其他可能性 (如贝叶斯准则、遗憾最小化) 进行比较和关联。6.4 节还给出了一个通用定理, 作为研究一大类算法的样本复杂性的基本理论机制, 其中包括 6.5 节中的基于模型的算法和 6.6 节中的模型无关算法。最后, 6.7 节总结了本章, 并概述了几个开放性问题。

6.2 预备知识

本章的研究对象是在线强化学习, 它比离线强化学习要更具有挑战性, 并且侧重于研究对于某些给定的 $\gamma \in (0, 1)$ 的 γ 减量累积回报最大化的学习器。

在本章中, 我们将环境建模为具有状态空间 \mathcal{S} , 动作空间 \mathcal{A} , 转换函数 T , 回报函数 R 和阻尼系数 γ 的马尔可夫决策过程 (MDP) $M = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$ 。本章中使用到了强化学习中的标准符号, 具体符号定义见第 1 章。特别地, 给定将 \mathcal{S} 映射到 \mathcal{A} 的策略 π , $V_M^\pi(s)$ 和 $Q_M^\pi(s, a)$ 分别是 MDP M 中 π 的状态和状态-动作价值函数; $V_M^*(s)$ 和 $Q_M^*(s, a)$ 是对应的最优价值函数。如果从上下文可以得知 M , 则可以省略 M , 并且价值函数将分别由 V^π 、 Q^π 、 V^* 和 Q^* 表示。最后, 我们使用 $\tilde{O}(\cdot)$ 作为抑制对数因子的普通大 O 符号的简写。通过以下定义

176

定义 6.1 学习器与环境之间的在线交互过程, 可以用 MDP $M = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$ 建模, 交互过程以离散时间步长 t 进行, $t = 1, 2, 3, \dots$;

- 1) 当学习器察觉到环境的当前状态 $s_t \in \mathcal{S}$, 然后采取动作 $a_t \in \mathcal{A}$ 。
- 2) 作为响应, 环境向学习器发送即时回报 $r_t \in [0, 1]^\ominus$, 并且移动到下一个状态 $s_{t+1} \in \mathcal{S}$ 。该转换由 MDP 的动态控制。具体地, r_t 的期望是 $R(s_t, a_t)$, 并且从分布 $T(s_t, a_t, \cdot)$ 随机地绘制下一个状态 s_{t+1} 。
- 3) 时钟方向: $t \leftarrow t + 1$ 。

此外, 假设最优价值函数的上限 V_{\max} , 使得对于所有 $s \in \mathcal{S}$, 都满足 $V_{\max} \geq V^*(s)$ 。由于假定 $R(s, a) \in [0, 1]$, 所以总是具有 $V_{\max} \leq 1/(1-\gamma)$, 尽管在许多情况下 $V_{\max} \ll 1/(1-\gamma)$ 。

在文献中已经提出了许多启发法。在实践中, 最流行的探索策略可能是 ϵ -贪婪策略: 学习器选择具有概率 ϵ 的随机动作, 否则根据其当前价值函数估计来选择最佳动作。虽然 ϵ -贪婪规则经常能保证对环境充分探索, 但它可能不是有效的, 因为探索在所有动作上均匀发生。有时发现诸如 soft-max、counter-based、recency-based、乐观初始化 (optimistic initialization) 和探索奖励 (exploration bonus) 之类的替代方案在经验上更为有效 [Sutton and Barto, 1998; Thrun, 1992]。例如, 使用间隔估计指导探索 [Kaelbling, 1993; Meuleau and Bourguine, 1999; Wiering and Schmidhuber, 1998]、信息增益 [Dearden, 1999]、MDP 特性 [Ratitch and Precup, 2002]、专家示范 (Abbeel and Ng, 2005) 等, 像这样的高级技术还有很

⊖ 并不一定要限制在 $r_t \in [0, 1]$, 因为任何有界回报函数可以被移位和重新缩放到范围 $[0, 1]$, 而不影响最优策略 [Ng et al, 1999]。

多, 这里仅举几例。

不幸的是, 尽管这些启发式技术在一些应用中工作良好, 但在其他应用中它们可能是低效的。下面是一个突出的例子 [Whitehead, 1991], 其中 ϵ -贪婪方法被证明是效率低下的。

示例 6.1 考虑具有 $N+1$ 个状态和 2 个动作的 MDP, 如图 6.1 所示。状态 1 是开始状态, 状态 G 是吸收目标状态。采取图中实线框中的动作将学习器传送到右边的状态, 而采取图中虚线框中的动作将学习器重置为

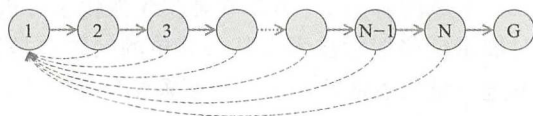


图 6.1 组合锁 [Whitehead, 1991]

开始状态 1, 并且它必须从 1 重新尝试到达目标状态。为简化展开, 除非 $s = G$ (此时有 $R(s, a) = 0$), 否则对于所有状态动作, 使用 $\gamma = 1$, 并且 $R(s, a) = -1$ 。

使用 ϵ -贪婪探索规则的学习器将在每个时间步长以至少 $\epsilon/2$ 的概率重置到开始状态。因此, 学习器总是选择实线框中的动作直到达到目标的概率是 $(1-\epsilon/2)^N$, 这意味着假定任何固定值 ϵ , 状态 N 的值在 N 中呈指数级的小。与此相反, 最优策略仅需要 N 个步骤, 本章后面描述的可证明有效的算法只需要少量的步骤 (N 的多项式数量级) 来找到目标以及最优策略。

这个例子表明, 一个差的探索规则可能是指数级低效的, 正如许多其他启发式所展现的那样。此外, 很多实践研究都证明, 低效探索可能会在实际应用中产生实质性影响, 而本章涉及的技术可以诱导出更智能的优于当前流行的那些启发式方法的探索方案 [Brunskill et al, 2009], Li and Littman, 2010; Nouri and Littman, 2009, 2010]。这些问题提出了设计有效的算法的关键需求, 而这正是本章的研究侧重点。

6.3 形式化探索效率

在开发可证明有效的算法之前, 必须明确定义有效的探索的概念意义。下面对某些探索的定义方法进行考察分析。

6.3.1 探索的样本复杂度和 PAC-MDP

在测量强化学习算法 A 的探索的样本复杂性 (或简称为样本复杂度) 时, 将复杂度与算法不选择接近最优行为的步骤数相关联是很自然的。为此, 我们将 A 视为将历史映射到动作的非固定策略。需要两个附加输入: $\epsilon < 0$ 和 $\delta \in (0, 1)$ 。精度参数 ϵ 控制我们对算法所要求的动作质量 (即我们期望该动作有多么接近最优行为)。置信参数 δ 表征我们期望的算法性能的确程度。随着两个参数减小, 算法期望获得更多环境的信息, 势必需要扩大探索范围和学习规模。

定义 6.2[Kadade, 2003] 令 $c_t = (s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_t)$ 是通过在 MDP M 中执行强化学习算法 A 直到时间步长 t 所产生的路径。该算法被视为非固定策略, 在时间 t 处表示为 A_t 。令 (r_t, r_{t+1}, \dots) 是在时间步长 t 之后由 A 接收的回报的随机序列。 S_t 处的状态值记作 $V^{A_t}(s_t)$, 它是减量累积回报 $r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$ 的期望。给定固定的 $\epsilon > 0$, A 的样本复杂度是执行时间步长的次数 τ , 在第 τ 时间步的策略 A_τ 不是 ϵ 最优的, 因为此时有: $V^{A_\tau}(s_t) \leq V^*(s_t) - \epsilon$ 。

算法 A 的样本复杂度是使用运行 A 和环境的学习器之间无限长的在线交互来定义的。如果我们把当 $V^{A_t}(s_t) \leq V^*(s_t) - \epsilon$ 不等式成立时的情况认为是“错误”的, 那么样本复杂度表示的就是 RL 算法在其整个运行期间所犯错误的总数。这里我们并不限定算法在与 MDP 无

限长的相互作用中出错的时间。这种自由对于 RL 是必要的，因为在随机 MDP 中，学习器不会对其将访问的状态序列具有绝对的控制权，因此错误可以在交互期间的任何时间发生，这取决于学习器何时随机转换到那些“坏”状态。直观地表述则是，我们喜欢具有较小样本复杂度的算法，这个事实启发我们得到以下探索效率的定义：

定义 6.3[Strehl et al, 2006a, b] 对于任意的 $\varepsilon > 0$ 且 $0 < \delta < 1$ ，如果 A 的样本复杂度以不小于 $1-\delta$ 的概率能通过某种基于 $(1/\varepsilon, 1/\delta, 1/(1-\gamma), |M|)$ 这些相关量的函数多项式界定，那么这样的算法 A 称为 MDP M 中的 PAC-MDP（可能大致正确的马尔可夫决策过程），其中 $|M|$ 表示 MDP M 的复杂性。通常， $|M|$ 与描述 M 的参数数量多项式相关；例如，它是有限 MDP 的状态和动作的数量，并且如果 MDP 可以以参数方式描述，那么它就是参数的数量。此外，如果算法的计算和样本复杂度都是多项式数量级的，则该算法称为“高效 PAC-MDP”算法。

定义 6.3 中涉及参数的讨论：

- 精度参数 ε 指定了我们希望算法实现的最优水平，当非稳定策略 A_t 不是 ε 最优时，会发生错误。在监督学习中，我们不能保证 $\varepsilon=0$ ，因为任何有限的样本集在随机时都不能准确地揭示 MDP 的完整动态。
- 置信参数 δ 衡量我们想要确定的算法探索的样本复杂度。在监督学习中，我们不能保证 $\delta=0$ ，因为总存在一个非零的（尽管它很小）概率，使得在这样的概率下，我们可以观察到的交互和样本并不能代表 MDP 的真实动态，从而误导学习器使之不能采取最优行为。
- 对 $1/(1-\gamma)$ 的依赖是强化学习中必要且常见的。在许多 RL 分析中，相关量 $1/(1-\gamma)$ 对于价值函数和策略起着大致有效的决策层的作用。决策层越大，样本复杂度也理应越大。
- MDP M 中的相关量 $|M|$ 通常用来表征 MDP（马尔可夫决策过程）有多大。它一般有一个自然的定义，并且当我们分析特定类别的 MDP 的样本复杂度时，这个相关量的值应当是提前已知的。

179

以上样本复杂度的定义首先用于 Rmax 的分析 [Kakade, 2003]，并将成为本章的重点。值得一提的是， E^3 算法家族的分析 [Kakade et al, 2003; Kearns and Koller, 1999; Kearns and Singh, 2002] 使用略有别于有效学习的定义。特别地，这些算法需要在多项式时间之后停止，并且以高概率输出最后状态的近似最优策略。我们在本章中的分析基本上也是一样的，唯一的区别在于我们在分析中去掉了混合时间参数，这让分析过程变得更为简单。

PAC-MDP 的概念是相当广义的，避免了诸如复位 [Fiechter, 1994, 1997]、并行采样预测 [Even-Dar et al, 2002; Kearns and Singh, 1999] 或转变矩阵的可达性（见下一小节）等这些假设。^①另一方面，如下所述，实现探索的小样本复杂度的标准在许多具有挑战性的问题中是易于处理的。

6.3.2 遗憾最小化

探索效率的另一个重要概念是遗憾（regret），采用平均回报标准可以更自然地定义这个

① 应该注意到当前的 PAC-MDP 分析强烈依赖于减量，而减量是使问题转变为有限区域问题的必要条件。这个问题将会在定理 6.1 的证明中澄明。

概念。通常，算法 A 的 T 步长遗憾是通过最优策略获得的 T 步累积回报与通过 RL 算法获得的 T 步累积回报之间的差，数学符号表示为 $L^A(T)$ 。其正式数学定义如下。

定义 6.4 给定一个 MDP M 和一个开始状态 s_1 ，算法 A 的 T 步返回值是学习器从状态 s_1 开始按照算法 A 获得累计回报，表示为：

$$R^A(M, s_1, T) \stackrel{\text{def}}{=} \sum_{t=1}^T r_t$$

平均每一步的回报通过下面公式给出

$$\rho^A(M, s_1) \stackrel{\text{def}}{=} \frac{1}{T} \mathbf{E}[R^A(M, s_1, T)]$$

类似的，可以定义静态策略 π ： $R^\pi(M, s_1, T)$ 和 $\rho^\pi(M, s_1)$ 。

根据一定的 M 的转换结构假设例如通信 [Puterman, 1994]，在静态策略 π^* 下 $\rho^{\pi^*}(M, s_1)$ 最大化，且最大值依赖于初始状态 s_1 ；定义 $\rho^* \stackrel{\text{def}}{=} \max_{\pi} \rho^\pi(M, s_1)$ 。则 T 步后算法 A 的遗憾由下式定义

$$L^A(T) \stackrel{\text{def}}{=} T\rho^* - R^A(M, s_1, T) \quad (6.1)$$

一般而言，回报最大化算法总是试图最小化遗憾。如果 $L^A(T) = o(T)$ ，则可以得出结论，学习器所遵循的非平稳策略 A 在累积回报方面与长期的最优策略太接近而导致难以区分， L^A 越小，算法探索和学习效率就越高。虽然遗憾最小化是在线强化学习的更直接的目标，但是具有子线性遗憾的算法一般可能不是 PAC-MDP[⊖]。另一方面，PAC-MDP 算法的遗憾有可能是非递减的，因为算法试图优化的目标是其正在遍历的状态的回报，而不是通过最优策略已经遍历过的状态的回报 [Jaksch et al, 2010, Footnote 4]。

然而，非平凡遗憾边界必须依赖于对基础 MDP 的转换结构的某些可达性假设：

示例 6.2 考虑以下包含 2 个动作的 MDP（马尔可夫决策过程）：如果学习器在 $t=1$ 时刻作出正确的动作，它将到达一个“天堂区” S_1 ，在这个区域里，访问任何状态的回报永远是 1，否则，它将抵达一个“地狱区” S_0 ，在这个区域中访问任何状态得到的回报都是 0。如果学习器无法从 S_0 中的状态转变到 S_1 中的状态，那么 $t=1$ 时刻学习器的次优动作将永远不能被弥补，因此，遗憾值的增长是线性的： $L^A(T) = \Theta(T)$ 。

因此，文献中的所有遗憾分析都依赖于各种可达性假设。特别地，假设转换矩阵不可约，可能出现渐近对数遗憾 [Burnetas and Katehakis, 1997; Tewari and Bartlett, 2008]： $L^A(T) \leq C \cdot \ln T$ ，其中 C 为常数。不幸的是， C 是取决于潜在的 MDP 的动态，所以事实上 C 可以任意大。

较弱的假设由 [Jaksch et al, 2010] 引入，其 UCRL2 算法具有的遗憾是 $\tilde{O}(D|S|\sqrt{A|T})$ ，其中定义了直径参数 D ，使得每个状态可以由平均最多 D 个步骤中的任何其他状态到达。最近，有研究学者对 [Bartlett and Tewari, 2009] 的 REGAL 算法的弱通信 MDP 应用了遗憾分析，产生了 $\tilde{O}(H|S|\sqrt{A|T})$ 的遗憾边界，其中 H 是最优价值函数的跨度。该界限改进了 UCRL2 的界限，因为对于所有弱通信 MDP，均有 $H \leq D$ 。

有趣的是，这些无遗憾的算法也实现了与许多 PAC-MDP 算法（6.4 ~ 6.6 节）一样的探索原则（称为“面对不确定性的乐观”）。例如，UCRL2 和 MBIE（6.5 节）使用间隔估计

⊖ 一个例子是一个单状态 MDP（也称为多武装匪徒）的探索计划，它在极限中贪婪，同时确保无限探索 [Robbins, 1952]：遗憾是关于 T 子线性的，但样本复杂度为 ∞ 。

来维持包含具有高概率的潜在 MDP 的一组 M 个 MDP，然后遵循 M 中最乐观模型的最优策略。主要区别在于它们分析 M 收缩的速度如何，以及它与遗憾以及样本复杂性是如何产生关联的。

最后，我们想强调遗憾的灵活性，作为可以在传统 MDP 之外使用的理论概念。例如，由具有强盗反馈的对抗在线学习中的竞争分析激发，最近的努力是导出强的遗憾边界，即使在 MDP 的奖励函数可以在每个步骤中以自适应对抗方式改变，给定对固定转变的完全知识概率（参见例如 [Neu et al, 2011] 及其中的参考文献）。

181

6.3.3 平均损失

在 6.3.2 节中的“天堂或地狱”的例子中，这个问题中的任何算法的样本复杂度是微不足道的 1，与不良的线性遗憾界限相反。显著差异突出了样本复杂性和遗憾之间的根本区别：前者是由算法访问的状态定义的，而后者是关于沿最优策略的轨迹的回报定义的。有人可能会认为，这种 MDP 很难解决，因为采取动作的错误无法在以后恢复，所以在遗憾方面的糟糕保证似乎比样本的复杂性更加符合问题的难度。然而，对于强化学习中的试错学习器来说犯错是不可避免的。因此，评估以历史为条件的算法似乎更自然。在这个意义上，“天堂或地狱”的 MDP 很容易解决，因为只有一个地方，学习器可能犯错误。

可以以不同的方式定义累积回报中的损失，以避免遗憾分析所需的可达性假设。一种可能性是使用重置（或情景任务）的假设，学习器可以周期性地回到相同的开始状态 [Fiechter, 1994, 1997]。但是，在大多数在线强化学习问题中，重置通常不可用。

另一种可能性称为平均损失 [Strehl and Littman, 2008a]，比较了学习器在实际访问的状态序列的累积回报上的损失：

定义 6.5 固定住 $T \in \mathbb{N}$ ，并让强化学习算法 A 连续执行 T 步。那么在时间步长 t 的瞬时损失 l_t 是最优状态值和累积折现回报之间的差为

$$l_t^{\text{def}} V^*(s_t) - \sum_{\tau=t}^T \gamma^{\tau-t} r_\tau$$

平均损失由下式定义

$$g(T)^{\text{def}} \frac{1}{T} \sum_{t=1}^T l_t$$

算法 A 就损失标准来看大概是近似正确的，如果满足对于任意 $\varepsilon > 0$, $\delta \in (0, 1)$ ，我们可以找到这样的 T ，它本身是关于 $(|S|, |A|, 1/(1-\gamma), 1/\varepsilon, 1/\delta)$ 这些量的多项式，且使得 $g(T) \leq \varepsilon$ 成立的概率至少为 $1-\delta$ 。

182

正如在样本复杂度的定义中所述，可达性假设是可以避免的，因为量 $g(T)$ 是定义在算法 A 实际访问过状态的序列之上的。在平均损失模型的学习能力相对较弱是因为它没有采用减少遗憾的方法 [Jaksch et al, 2010, Footnote 4]。更进一步地，可以发现任何 PAC-MDP 算法在平均损失标准上都是大致近似正确的 [Strehl and Littman, 2008a]。

6.3.4 贝叶斯框架

在贝叶斯设置中，学习器可以假设在指定 MDP（包括转换概率和回报函数）的参数上的先前分布，并且在与环境的在线交互期间使用贝叶斯规则维持后验分布。然后，我们可以要求贝叶斯最优策略，最大化相对于后验分布的预期累积回报。在某种意义上，在这个设置中

不需要明确的探索，因为 MDP 模型中的不确定性完全由定义的后验分布捕获。因此，学习器可以简单地遵循贝叶斯最优策略 [Duff, 2002]。

贝叶斯方法的一个主要限制是它们的高计算复杂性。一般来说，计算贝叶斯最优策略通常是棘手的，使得近似不可避免（参见，例如 [Duff, 2002]）。它仍然是一个积极的研究领域以开发有效的算法来近似贝叶斯最优策略 [Poupart et al, 2006; Kolter and Ng, 2009]。

虽然计算昂贵，但贝叶斯强化学习具有能够使用模型的先验分布来指导探索的重要益处。有了适当的先验，贝叶斯探索可以有出色的性能。相比之下，具有小样本复杂度或低遗憾的算法通常被开发用于处理最坏情况的场景，并且在实践中常常是保守的。然而，贝叶斯方法的好处带来了一个代价：它们通常缺乏强有力的理论保证，尤其是在先前分配被错误指定时。例如，贝叶斯算法通常不是 PAC-MDP，或者甚至可能最终收敛到一个次优策略；相关实例可以在 [Kolter and Ng, 2009, Theorem 2; Li, 2009, Example 9] 中找到。

最后，我们注意到，有可能组合一些贝叶斯和 PAC-MDP 方法的技术，以融合两者的优势。例如，为 PAC-MDP 算法开发的分析工具用于推导 BEB 算法，其近似贝叶斯最优策略，除了多项式级数量的步骤下的情况 [Kolter and Ng, 2009]。作为另一个例子，已知状态-动作对与模型的后验分布结合以产生能够使用关于 MDP 模型的先验知识 [Asmuth et al, 2009] 的随机化 PAC-MDP 算法 (BOSS)。

6.4 通用 PAC-MDP 定理

本节给出了一个通用定理，用来证明一类在线强化学习算法的多项式样本复杂度。这个结果将作为一个基本工具为本章后续研究基于模型和无模型的方法服务。

考虑存在一个强化学习算法 A ，它维持着一个估计的状态-动作价值函数 $Q(\cdot, \cdot)$ ，现在，令 Q_t 表示学习器的第 t 个动作之前的 Q 的即刻值。我们称 A 是贪婪的，如果它始终选择一个使得其当前价值函数估计最大化的动作，即 $a_t = \arg \max_{a \in A} Q_t(s_t, a)$ ，其中 s_t 是该学习器到达的第 t 个状态。为了以后便于表述，我们定义 $V_t(s) \stackrel{\text{def}}{=} \max_{a \in A} Q_t(s, a)$ 。至于我们的讨论，需要提及两个重要的定义，而在论述这两个定义之前需要提及一些定义中会用到的符号表述，其一是任意的状态动作集，表述为 $K \subseteq S \times A$ ，其中 K 是任意定义的，一般理解成一个已经被学习器所获知且不必再去探索的子集。最后是状态 s ，如果 $(s, a) \in K$ 对于所有的 $a \in A$ 成立，那么可以称 s 是“已知的”。

定义 6.6 我们定义 E_t 为 K 中的“逃离事件”，表示算法 A 在第 t 时间步长遭遇到的状态-动作对中存在部分不属于 K ，即 $(s, a) \notin K$ 。

定义 6.7 给定一个 MDP $M = \langle S, A, T, R, \gamma \rangle$ 和一个状态-动作价值函数 Q ，关于 K 的已知状态动作 MDP 标记为 M_K ，它是一个有着相同状态动作集合但有不同的回报和转换函数的 MDP：

$$T_K(s, a, s') = \begin{cases} T(s, a, s') & , (s, a) \in K \\ \prod(s' = s) & , \text{其他} \end{cases} \quad (6.2)$$

$$R_K(s, a) = \begin{cases} R(s, a) & , (s, a) \in K \\ (1 - \gamma)Q(s, a) & , \text{其他} \end{cases} \quad (6.3)$$

其中 $\prod(s' = s)$ 是指示函数。如果我们在公式 (6.2) 和公式 (6.3) 的等号右边替换真正的动态 T 和 R ，而采用 \hat{T} 和 \hat{R} ，那么最终的 MDP，记作 \hat{M}_K ，称为关于 K 经验已知的状态动作

MDP。直观地说,只要 Q 是乐观的 (即 $Q(s, a) \geq Q^*(s, a)$ 对于所有的状态动作对 (s, a) 都成立),那么已知的状态动作 MDP M_K 就是真实 MDP 的一个乐观模型。更进一步地说,这两个 MDP 的动态在 K 中的状态动作集合上是保持一致的。现在,我们考虑以下两个例子:

1) 如果一个学习器按照任何规则都需要太多的步骤才能从当前状态转换到某个未知的状态,那么基于 γ 减量的特性,其探索的价值就很低,换句话说,在状态 s 下所有未知状态都是必然和最优行为选择无关的,因此学习器就只需要按照 M_K 的最优策略选择动作,这样的话,至少可以保证学习器能在状态 s 接近最优的模型 M 。

2) 另一方面,如果一个未知状态和当前所处状态很接近, M_K 的构建 (假设 Q 是乐观的) 将会激励学习器去探索未知的状态 (也就是“逃离”事件),那么当前状态下的策略就可能不是近似最优的。

184

因此,第二种情形发生的次数直接与算法的复杂度关联。已知状态-动作 MDP 是优雅地平衡探索和开发的关键概念。

上述直观的描述可以通过下列一般性原理形式化表达 (Li, 2009),该原理仅对原始结果进行了小幅度的提升。它为本章后续所有样本复杂度分析提供了一个统一的基础。

定理 6.1 令 $A(\varepsilon, \delta)$ 为一个算法,该算法采用 ε 和 δ 作为输入 (加上其他的算法指定的输入),并采用贪婪策略,即每一步的动作选择仅仅依赖在当前第 t 时间步长上的估计价值函数,我们标记这个函数在时刻 t 为 Q_t 。假设在每一个时间步长上,都存在一个状态动作集合 K_t ,它只依赖学习器直到当前时刻 t 的所有历史。如果在第 t 时间步长上,状态-动作对的值没有发生变化并且没有发生“逃离”事件 E_t ,我们显然可以假设 $K_t = K_{t+1}$ 。令 M_{K_t} 表示已知状态-动作 MDP (根据定义 6.7 中的 K_t 和 Q_t 来定义),同时令 π_t 表示基于估计价值函数 Q_t 的贪婪策略。假设对于任何给定的输入 ε 和 δ ,下列条件以至少 $1-\delta/2$ 的概率在任何时间步长内都保持成立:

1) (乐观主义) $V_t(s_t) \geq V^*(s_t) - \varepsilon/4$ 。

2) (精确度) $V_t(s_t) - V_{M_{K_t}}^{\pi_t}(s_t) \leq \varepsilon/4$ 。

3) (有界变化) 动作值估计更新的总次数加上来自 K_t 的逃离事件可能发生的次数的乘积,是由函数 $\zeta(\varepsilon, \delta, |M|)$ 界定。

那么,至少以 $1-\delta$ 的概率,样本复杂度 $A(\varepsilon, \delta)$ 等于下式:

$$O\left(\frac{V \max}{\varepsilon(1-\gamma)} \left(\zeta(\varepsilon, \delta, |M|) + \ln \frac{1}{\delta} \right) \ln \frac{1}{\varepsilon(1-\gamma)}\right)$$

证明 (此处的证明仅为粗略的论证) 该证明包含两个主要部分。第一部分,令 W 表示这样一种事件,从状态 s_t 开始遵循非平稳策略 A_t 并进行了 $H = \frac{1}{1-\gamma} \ln \frac{4}{\varepsilon(1-\gamma)}$ 个时间步长后,发生了以下事件中的任何一个: 1) 某些状态-动作对价值函数估计被改变了; 2) 一个 K_t 中的“逃离”事件发生了,也就是说学习器遍历过的状态-动作对 (s, a) 中存在部分不属于 K_t 。上诉乐观主义和精度条件暗示着一个分析的启发式结果,称为“隐式的探索和开发”引理 [Kakade, 2003; Kearns and Singh, 2002]:

$$V^{A_t}(s_t) \geq V^*(s_t) - 3\varepsilon/4 - \Pr(W) V_{\max}$$

换句话说,如果 $\Pr(W) < \frac{\varepsilon}{4V_{\max}}$,则 A_t 是在状态 s_t 下 ε -最优的 (开发); 否则学习器会以至少 $\frac{\varepsilon}{4V_{\max}}$ 的概率经历事件 W (探索)。

证明的第二部分给出了 $\Pr(W) \geq \frac{\varepsilon}{4V_{\max}}$ 发生的次数的界限。为了证明这点，我们将学习器经历的所有状态-动作对形成的整个轨迹划分成长度为 H 的碎片，每一个碎片都从状态 s_{iH+1} 开始，其中 $i \in \mathbb{N}$ ，所有碎片的状态可以表示为 $(s_1, \dots, s_H), (s_{H+1}, \dots, s_{2H}), \dots$ 。然而，我们将学习器从状态 s_{iH+1} 开始并遵循算法 A 动作的 H 步子轨迹视为伯努利实验 (Bernoulli trial)。如果事件 W 发生了，那么我们就认为伯努利实验成功了。显然，这些伯努利实验的结果都是相互独立的，仅仅依赖于各自轨迹的初始状态。由于每一个伯努利实验都以至少 $\frac{\varepsilon}{4V_{\max}}$ 的概率成功，因此总的成功实验数量最多是 $\zeta(\varepsilon, \delta, |M|)$ (这一点是由上面给出的有界变化条件得出)。可以证明，根据 Azuma 的不等式，在至少为 $1-\delta/2$ 的概率下，有最多 $\frac{8HV_{\max}}{\varepsilon} \left(\zeta(\varepsilon, \delta) + \ln \frac{2}{\delta} \right)$ 个时间步长是满足 $\Pr(W) \geq \frac{\varepsilon}{4V_{\max}}$ 不等式的。

最后定理应用了下述两个限制的一种联合形式，即任何一个事件发生的概率最大都是 $\delta/2$ ：1) 上面提及的三个条件可能失效；2) 上面的证明中的第二部分也可能失效。

定理 6.1 是一个强大的理论工具，它提供了一个便利的方法来证明一个算法是 PAC-MDP，正如接下来的两小节提到的基于模型和无模型算法所展示的那样。

6.5 基于模型的方法

本节介绍基于模型的 PAC-MDP 算法。这些算法通常明确地估计未知 MDP 的参数，然后计算估计的 MDP 模型的最优策略。仿真引理 (Simulation Lemma, 请参考 [Kearns and Singh, 2002]) 指出，具有类似动力学的 MDP 具有类似的价值函数，因此要确保在真实的 MDP 中策略将接近最优，只需满足让估计的 MDP 模型不断学习直到达到足够的精确度。

基于模型的算法通常需要以子例程的形式来求解 MDP，因此计算量非常大。然而，通过维护关于该问题的更多信息，基于模型的算法在实践中常常表现得更有效 (参见例如 [Moore and Atkeson, 1993])。事实上，第一个 PAC-MDP 算法包括 E^3 [Kearns and Singh, 2002] 和 Rmax [Brafman and Tenen Holtz, 2002] 明确区分了“已知状态”集合，其中精确的回报和转换概率都是可以从“未知状态”集合中预测到的。结果表明，这种技术可以扩展到基于一个新型学习模型的一般 MDP 上 (详见 6.5.2 节)。

6.5.1 Rmax

算法 12 给出了一个完整的 Rmax 的伪代码。Rmax 的设计中使用了两个关键思想。一个是已知状态之间的区别，这里的状态指的是其中转换分布和回报可以从观察到的转换和未知状态中精确地推断出来的那部分。另一个是称为“面对不确定性的乐观主义”的探索原则 [Brafman and Tenen Holtz, 2002]。

Rmax 第一个重要的组成部分是已知状态动作的概念，假设一个动作 $a \in A$ 在状态 $s \in S$ 中被执行了 m 次。令 $r[i]$ 和 $s'[i]$ 分别表示第 i 次观测到的回报和下一状态，相对地，对于 $i=1, 2, \dots, m$ ，一个关于状态动作 (s, a) 的回报和转换函数极大似然估计按如下的形式给出：

$$\hat{T}(s, a, s') = |\{i | s'[i] = s'\}| / m, \quad \forall s' \in S \quad (6.4)$$

$$\hat{R}(s, a) = (r[1] + r[2] + \dots + r[m]) / m \quad (6.5)$$

0: **Inputs:** $\mathbf{S}, \mathbf{A}, \gamma, \varepsilon, \delta, m$.

1: Initialize counter $c(s, a) \leftarrow 0$ for all $(s, a) \in \mathbf{S} \times \mathbf{A}$.

2: Initialize the empirical known state-action MDP $\hat{M} = (\mathbf{S}, \mathbf{A}, \hat{T}, \hat{R}, \gamma)$:

$$\hat{T}(s, a, s') = \mathbb{I}(s' = s), \quad \hat{R}(s, a) = V_{\max}(1 - \gamma).$$

3: **for all** timesteps $t = 1, 2, 3, \dots$ **do**

4: Compute the optimal state-action value function of \hat{M} , denoted Q_t .

5: Observe the current state s_t , take a greedy action $a_t = \arg \max_{a \in \mathbf{A}} Q_t(s_t, a)$, receive reward r_t , and transition to the next state s_{t+1} .

6: $c(s_t, a_t) \leftarrow c(s_t, a_t) + 1$

7: **if** $c(s_t, a_t) = m$ **then**

8: Redefine $\hat{T}(s_t, a_t, \cdot)$ and $\hat{R}(s_t, a_t)$ using Equations 6.4 and 6.5.

算法 12 Rmax

直观地看, 随着 m 的增大, 我们预期 $R(s, a)$ 和 $T(s, a, s')$ 这些估计几乎一定会收敛到各自的真实值。当 m 足够大 (大到足够让算法能快速收敛到要求的精确度) 时, 状态动作对 (s, a) 就可以称为是“已知”的, 这是因为此时我们得到了在这个状态上的回报和转换概率的精确估计, 否则 (s, a) 是“未知”的。如果令 K_t 表示在第 t 时间步长时已知状态-动作对的集合 (不包括时刻 t 的状态和选择的动作构成的数组)。显然, 对于任意 t 都有 $K_t = \emptyset$ 且 $K_t \subseteq K_{t+1}$ 。

Rmax 第二个关键的部分是显式地处理探索问题。根据仿真引理, 一旦所有的状态-动作对都变为已知, 学习器就能够以高精度逼近未知的 MDP 并因此获得近似最优的表现。因此应该鼓励学习器去探索未知的状态-动作对。为了实现这个目标, 需要对未知状态-动作对的函数估计赋予一个最大的值 V_{\max} , 即对于所有的 t , 在第 t 时间步长上, 都令 $Q_t(s, a) = V_{\max}$, 如果 $(s, a) \notin K_t$ 。在算法执行时, 若动作 a_t 在状态 s_t 执行了 m 次, 则 $K_{t+1} = K_t \cup \{(s_t, a_t)\}$ 。显然, 在第 t 时间步长上通过 Rmax 求解出来的经验状态-动作 MDP 恰好和定义 6.7 中 (关于 K_t 和 Q_t 的部分) 的经验已知状态动作 MDP 重合。

定理 6.1 给出了一个简便的方式来证明 Rmax 就是 PAC-MDP。尽管当前已经存在完整的证明 [Kakade, 2003; Strehl et al, 2009], 但我们这里只给出该证明的一个凸显分析中的重要步骤的粗略版本。

定理 6.2 Rmax 是具有以下样本复杂度的 PAC-MDP:

$$\tilde{O}\left(\frac{|\mathbf{S}|^2 |\mathbf{A}| V_{\max}^3}{\varepsilon^3 (1-\gamma)^3}\right)$$

证明 (粗略) 只要验证 Rmax 满足定理 6.1 中的三个条件就足够了。使用切尔诺夫-霍夫丁 (Chernoff/Hoeffding) 不等式, 我们可以表明公式 (6.4) 和公式 (6.5) 中的最大似然估计对于大的 m 是非常准确的: 准确地说, 以下等式以至少 $1-\delta$ 的概率成立,

$$|\hat{R}(s, a) - R(s, a)| = O(\varepsilon(1-\gamma))$$

$$\sum_{s' \in \mathbf{S}} \hat{T}(s, a, s') - T(s, a, s') = O\left(\frac{\varepsilon(1-\gamma)}{V_{\max}}\right)$$

对于任何给定的 (s, a) , 和某些 $m_0 = \tilde{O}\left(\frac{|\mathbf{S}| V_{\max}^2}{\varepsilon^2 (1-\gamma)^2}\right)$ 只要满足 $m \geq m_0$ 即可。有了这些对回报

和转换概率的高精确度的估计, 定理 6.1 中的乐观主义和精度条件立即就能通过仿真引理推导出来。更进一步地, 因为任何未知的状态-动作对都最多只能重现 m 次 (超过这个次数, 改状态-动作对就会变成已知的), 定理 6.1 中的有界变化条件也同时成立, 并且有

$$\zeta(\varepsilon, \delta, |M|) = |S| |A| m = \tilde{O}\left(\frac{|S|^2 |A| V_{\max}^2}{\varepsilon^2 (1-\gamma)^2}\right)$$

Rmax 的样本复杂度也立刻就能知道了。

Rmax 算法和其样本复杂度分析可以从多个方面进行提升。例如, 状态-动作的已知属性的二值设定可以用间隔估计 (interval estimation) 来平滑地量化极大似然估计的不确定性, 这就是 MBIE 算法 [Strehl and Littman, 2008a]。另外, 我们可以采用一个非固定取值的乐观的价值函数以取代固定常数值 V_{\max} , 以此优化样本复杂度的界限确定 [Strehl et al, 2009; Szita and Lőrincz, 2008]。更重要的是, 我们将在下一小节中展示如何在一个新的监督学习模型的帮助下拓展 Rmax, 将其由有限的 MDP 拓展为通用的和潜在无限的 MDP。

最后, 我们注意到, 最后提出了一个来自 Rmax 的变体算法 MoRmax, 它的样本复杂度是 $\tilde{O}\left(\frac{|S| |A| V_{\max}^2}{\varepsilon^2 (1-\gamma)^4}\right)$ 。和 Rmax 不同的是, 对于一个状态-动作对而言, 一旦这个状态-动作对被确定为已知, Rmax 的转换概率和回报的估计就立即被固定下来。但 MoRmax 有时会重新估计这些量: 如果收集到了新的长度为 m 的样本集, MoRmax 算法就会重新估算转换概率和回报。尽管这个算法和其相关的分析更加复杂, 但它的样本复杂度方面相比 Rmax 而言有明显的改进, 改进的效果和 $|S|$ 、 $1/\varepsilon$ 以及 $1/V_{\max}$ 相关, 代价是多了个因子 $1/(1-\gamma)$ 。另值得一提的是, 该算法所依赖的 $|S|$ 的边界是目前要求最低的 [Li, 2009]。

6.5.2 Rmax 的泛化

Rmax 算法并不适用于连续状态 MDP, 而这种 MDP 在某些应用中是很常见的。另外, 该算法的样本复杂度显式依赖于 MDP 中状态-动作对的个数, 这暗示着该算法将会在那些状态或动作空间很大的特定的 MDP 上表现效率低下。在监督学习中, 泛化一般都是用来避免对状态-动作对进行显式地枚举, 但这会导致探索变得更加具有挑战性。下面我们首先介绍一个监督学习的新模型, 该模型可以帮助我们拓展 Rmax 成为任意的 MDP。

6.5.2.1 明确自己所知道的

正如前面解释过的, Rmax 一个关键属性是根据状态-动作对动态预测的不确定性区分该状态-动作对是已知还是未知, 这个知识提供了有效的机制使得探索变得高效。受到这一点的启发, “明确自己所知道的” (Knows What It Knows, KWIK) 框架被提出来, 用来捕获不确定性估计的量化本质 [Li et al, 2011]。

一个 KWIK 问题由有 5 个参数确定: $P = \langle \mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{H}, h^* \rangle$, 其中 \mathbf{X} 是一个输入集合; \mathbf{Y} 是一个输出集合; \mathbf{Z} 是一个观测集合; $\mathbf{H} \subseteq \mathbf{Y}^{\mathbf{X}}$ 是一类假定的从 \mathbf{X} 到 \mathbf{Y} 的映射; 且 $h^* \in \mathbf{Y}^{\mathbf{X}}$ 是一个未知的目标函数。我们称一个 KWIK 问题是“可实现的”, 如果它满足 $h^* \in \mathbf{H}$; 否则, 称它为“不可知的”。

KWIK 学习过程中包含两个部分, 首先, 学习器执行学习算法并做出预测; 其次, 代表 KWIK 学习问题实例的环境提供给学习器它所需要的输入和取样。这两部分按照以下协议进行交互。

定义 6.8 一个 KWIK 学习进程按照以下步骤执行 (图 6.2):

1) 输入集合 \mathbf{X} , 输出集合 \mathbf{Y} , 取样集合 \mathbf{Z} , 假定类 \mathbf{H} , 精确参数 $\varepsilon > 0$, 置信度 $\delta \in (0,1)$ 作为学习器和环境的共同已知参数。

2) 环境秘密地敌对地选择一个目标函数 $h^* \in \mathbf{Y}^{\mathbf{X}}$ 。

3) 在第 $t=1,2,3,\dots$ 时间步长上,

- 环境任意选择一个输入 $x_t \in \mathbf{X}$, 并通知学习器。目标值 $y_t = h^*(x_t)$ 对于学习器来说是未知的。注意 x_t 的选取可能依赖交互历史。
- 学习器预测一个输出 $\hat{y}_t \in \mathbf{Y} \cup \{\perp\}$, 其中的 \perp 是一个特殊的符号, 它不属于 \mathbf{Y} 。我们称 \hat{y}_t 是合理的, 如果满足 $\hat{y}_t \neq \perp$ 。
- 如果 $\hat{y}_t = \perp$, 学习器对输出进行一次取样 $z_t \in \mathbf{Z}$ 。在确定性的情形下, $z_t = y_t$ 。在随机性的情形下, z_t 和 y_t 的关系由问题给定。比如, 在伯努利实验的情形下, $\mathbf{Y} = [0,1]$ 且 $\mathbf{Z} = \{0,1\}$, 此时, $z_t = 1$ 以 y_t 的概率成立, 其余情况则为 0。

定义 6.9 令 $\mathbf{H} \subseteq \mathbf{Y}^{\mathbf{X}}$ 表示一个假定类。我们称 \mathbf{H} 是基于 KWIK 可学习的, 如果存在一个算法 A 具有以下性质: 对于任意的 $\varepsilon > 0$ 和 $\delta \in (0,1)$, 算法根据以上 KWIK 协议的一次完整执行过程以至少为 $1-\delta$ 的概率满足以下两个要求,

1) (精确度要求) 如果 $h^* \in \mathbf{H}$, 则所有非 \perp 预测必须是 ε 准确的; 也即, 只要满足 $\hat{y}_t \neq \perp$, 任何时候都有 $|\hat{y}_t - y_t| < \varepsilon$ 成立^①。

2) (样本复杂度要求) 在算法的整个运行过程中的所有预测的 \perp 的总数标记为 $B(\varepsilon, \delta, \dim(\mathbf{H}))$, 它的边界通过一个由 $1/\varepsilon$ 、 $1/\delta$ 和 $\dim(\mathbf{H})$ 构成的多项式函数确定, 并且其中的 $\dim(\mathbf{H})$ 是一个预定义的非负取值函数, 它起着衡量 \mathbf{H} 的维度或复杂度。

189

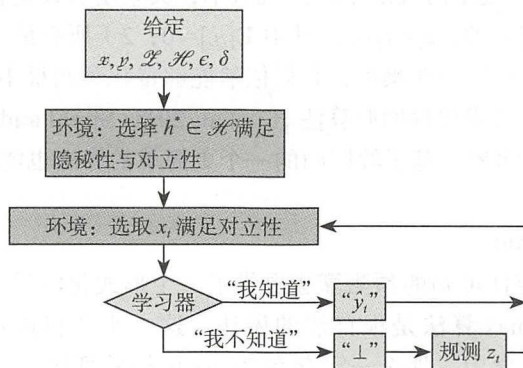


图 6.2 KWIK 协议

我们称具有以上两个性质的算法 A 是一个 KWIK 算法, 称 $B(\varepsilon, \delta, \dim(\mathbf{H}))$ 是算法 A 的一个 KWIK 边界。更进一步地, 我们称 \mathbf{H} 是高效 KWIK 可学习的, 如果算法 A 在满足精确度和样本复杂度的要求之外, 同时满足每一步的算法时间复杂度是一个由 $1/\varepsilon$ 、 $1/\delta$ 和 $\dim(\mathbf{H})$ 构成的多项式函数。

尽管 KWIK 是一个比以往模型 (如概率近似正确模型 [Valiant, 1984] 和误差边界模型

① 此处, $|\hat{y} - y|$ 可以是任何度量 \hat{y} 与 y 之间差异或举例的非负取值函数。为了让表述更直观些, 我们采用广义的操作符 “—”, 它在这里不局限于算术意义上的减法含义。比如: 1) 在 $\mathbf{Y} \subseteq \mathbb{R}$ 的特殊情况下, $|\cdot|$ 被理解成绝对值; 2) 当 $\mathbf{Y} \subseteq \mathbb{R}^d$ 时, $|\cdot|$ 则表示向量的模; 3) 当 y 和 \hat{y} 是概率分布, 则 $|\cdot|$ 表示总体变化。

[Littlestone, 1987]) 更难学习的模型, 但有不少假定类被证明是高效 KWIK 可学习的。以下, 我们通过 [Li et al, 2001] 回顾一些具有代表性的例子。

例子 6.3 考虑学习一个确定概率函数 $f(x) \stackrel{\text{def}}{=} x \cdot \theta^*$ 以确定其未知参数向量 $\theta^* \in \mathbb{R}^d$ 的问题: 输入 $x \in \mathbb{R}^d$ 是一个维度为 d 的向量, 并且输出和取样都是 $y = z = f(x)$ 。以 KWIK 方式学习这样的线性函数, 我们可能需要维持一个包含训练样本的集合 \mathbf{D} , 它一开始初始化为空集, 在学习的过程中, 假设当前第 t 次输入的是 x_t , 令

$$\mathbf{D} = \{(v_1, f(v_1)), (v_2, f(v_2)), \dots, (v_k, f(v_k))\}$$

为当前的训练样本集合。算法首先检查 x_t 是否和 \mathbf{D} 中的历史输入样本线性无关, 如果线性无关, 那么算法预测 \perp , 取样输出 $y_t = f(x_t)$, 然后扩展训练样本集合: $\mathbf{D} \leftarrow \mathbf{D} \cup \{(x_t, y_t)\}$ 。否则存在 k 个实数 a_1, a_2, \dots, a_k 满足 $x_t = a_1 v_1 + a_2 v_2 + \dots + a_k v_k$ 。对于后一种情况, 我们可以精确的预测 $f(x_t)$ 的值: $f(x_t) = a_1 f(v_1) + a_2 f(v_2) + \dots + a_k f(v_k)$ 。另外, 因为 \mathbf{D} 可能包含至多 d 个线性无关输入, 算法的 KWIK 边界是 d 。

190

例子 6.4 作为一个随机问题的例子, 考虑学习一个 n 元多项分布问题, 该分布 p 由 n 个总和为 1 的非负数确定: $p = (p_1, p_2, \dots, p_n)$, 其中 $p_i \geq 0$ 且 $\sum_{i=1}^n p_i = 1$ 。由于要学习一个单一的多项分布 p^* , 因此输入之间是无关的, 输出一直是 p^* 。总变化可以通过 $|\hat{p} - p| \stackrel{\text{def}}{=} \frac{1}{2} \sum_{i=1}^n |\hat{p}_i - p_i|$ 度量。对于这个问题, 我们可以预测 \perp 和最初的 m 步获取的随机样本, 然后一直预测极大似然估计 \hat{p} , 且只要 $m \geq \frac{2n}{\varepsilon^2} \ln \frac{2n}{\delta}$, \hat{p} 就是精确的预测值 [Kakade, 2003, Lemma 8.5.5]。这个算法称为骰子学习 (是根据估计掷一个骰子的偏差而命名的), 它的 KWIK 边界是 $\tilde{O}(n/\varepsilon^2)$ 。

例子 6.5 我们现在考虑例子 6.3 中的随机版本, 其余所有设定保持不变, 除了: 1) 取样过程中加入白噪声进行腐蚀, $z_t = y_t + \eta_t$, 其中 $\mathbf{E}[\eta_t] = 0$; 2) 所有量 (包含 x_t 、 θ^* 和 η_t) 都是 l_2 范数有界的。尽管, 取样中存在噪声, 我们依然能够量化预测最小方差估计的误差。利用一个覆盖参数, 可以证明带噪声线性回归算法 [Li et al, 2011; Strehl and Littman, 2008b] 具有一个值为 $\tilde{O}(d^3/\varepsilon^4)$ 的 KWIK 边界。基于岭回归的一个更简单的算法也能得到相似的结论 [Walsh et al, 2009]。

6.5.2.2 KWIK-Rmax

KWIK 为研究不确定性可知的预测算法提供了一个形式化的学习模型。结合学习 MDP 的 KWIK 算法和基本 Rmax 算法是很自然的做法。这一小节将展示主要的结果: 如果一类 MDP 可以通过 KWIK 学习, 那么一定存在 Rmax 风格的算法, 且该算法是该类 MDP 的 PAC-MAP 算法。

我们首先定义一类 MDP 的 KWIK 可学习性, 它是由仿真引理启发而来的。对任何集合 A , 我们记 \mathbf{P}_A 是一个 A 上的概率分布集合。

定义 6.10 固定住状态空间 \mathbf{S} , 动作空间 \mathbf{A} , 阻尼系数 γ 。

1) 定义 $\mathbf{X} = \mathbf{S} \times \mathbf{A}$, $\mathbf{Y}_T = \mathbf{P}_S$, 和 $\mathbf{Z}_T = \mathbf{S}$ 。令 $\mathbf{H}_T \subseteq \mathbf{Y}_T^{\mathbf{X}}$ 为一个 MDP 的转换函数。如果根据定义 6.9 中的精确度要求 $|\hat{T}(\cdot|s, a) - T(\cdot|s, a)|$ 按照 l_1 范式距离来定义, 如下:

$$\begin{cases} \sum_{s' \in \mathbf{S}} |\hat{T}(s'|s, a) - T(s'|s, a)| & , \mathbf{S} \text{ 是可数的} \\ \int_{s' \in \mathbf{S}} |\hat{T}(s'|s, a) - T(s'|s, a)| ds' & , \text{其他} \end{cases}$$

191

则我们称 \mathbf{H}_T 是 (高效) KWIK 可学习的

2) 定义 $\mathbf{X} = \mathbf{S} \times \mathbf{A}$ 和 $\mathbf{Y}_R = \mathbf{Z}_R = [0, 1]$, 令 $\mathbf{H}_R \subseteq \mathbf{Y}_R^{\mathbf{X}}$ 是一个 MDP 的回报函数集合。如果根据定义 6.9 的精确度要求, 把 $|\hat{R}(s, a) - R(s, a)|$ 理解为一个绝对值; 则我们称 \mathbf{H}_R 是 (高效) KWIK 可学习的。

3) 令 $\mathbf{M} = \{\mathbf{S}, \mathbf{A}, T, R, \gamma | T \in \mathbf{H}_T, R \in \mathbf{H}_R\}$ 为一类 MDP。如果 \mathbf{H}_T 和 \mathbf{H}_R 均是 (高效) KWIK 可学习的, 则我们称 \mathbf{M} 是 (高效) KWIK 可学习的。

简单地说, 如果一类 MDP 的转换和回报函数都是 KWIK 可学习的, 那么我们定义该类 MDP 是 KWIK 可学习的。如下所示, MDP 类的 KWIK 可学习性足够保证一个 Rmax 风格的算法具有 PAC-MDP 的性质。

一个通用型的算法 (称为 KWIK-Rmax) 在算法 13 中已经给出。和 Rmax 类似, 它显式地区分已知的状态-动作对和未知的状态-动作对。然而, 它是通过两个 KWIK 算法 A_T 和 A_R 来实现的, 其中 A_T 用来实现转换函数学习, 而 A_R 用于回报函数学习。已知的状态-动作对的集合很显然是 KWIK 算法可以合理预测的。根据定义, 这些预测必须是高度精确的以便于能够可靠地使用它们。在 Rmax 中, 为未知状态-动作对赋值一个乐观的数值 V_{\max} , 用来激励学习器进行探索。最后, 在解一个经验已知状态动作 MDP 时算法允许存在一个近似误差。这很重要, 因为当 MDP 非常大或者是连续的时候, 精确的规划是代价昂贵的甚至是不可能做到的。

```

0: Inputs:  $\mathbf{S}, \mathbf{A}, \gamma, V_{\max}, A_T$  (with parameters  $\varepsilon_T, \delta_T$ ),  $A_R$  (with parameters  $\varepsilon_R, \delta_R$ ),  $\varepsilon_P$ 
1: for all timesteps  $t = 1, 2, 3, \dots$  do
2:   // Update the empirical known state-action MDP  $\hat{M} = \langle \mathbf{S}, \mathbf{A}, \hat{T}, \hat{R}, \gamma \rangle$ 
3:   for all  $(s, a) \in \mathbf{S} \times \mathbf{A}$  do
4:     if  $A_T(s, a) = \perp$  or  $A_R(s, a) = \perp$  then
5:        $\hat{T}(s' | s, a) = \begin{cases} 1 & \text{if } s' = s \\ 0 & \text{otherwise} \end{cases}$  and  $\hat{R}(s, a) = (1 - \gamma)V_{\max}$ 
6:     else
7:        $\hat{T}(\cdot | s, a) = A_T(s, a)$  and  $\hat{R}(s, a) = A_R(s, a)$ 
8:   Compute a near-optimal value function  $Q_t$  of  $\hat{M}$  such that  $|Q_t(s, a) - Q_M^*(s, a)| \leq \varepsilon_P$ 
   for all  $(s, a)$ , where  $Q_M^*$  is the optimal state-action value function of  $\hat{M}$ .
9:   Observe the current state  $s_t$ , take action  $a_t = \arg \max_{a \in \mathbf{A}} Q_t(s_t, a)$ , receive reward  $r_t$ , and transition to the next state  $s_{t+1}$ .
10:  if  $A_T(s_t, a_t) = \perp$  then
11:    Inform  $A_T$  of the sample  $(s_t, a_t) \rightarrow s_{t+1}$ .
12:  if  $A_R(s_t, a_t) = \perp$  then
13:    Inform  $A_R$  of the sample  $s_t \rightarrow r_t$ .

```

算法 13 KWIK-Rmax

算法 13 主要是用来简化阐述过程并且强调基础算法组成部分的。如果想要实现该算法, 那么以下几点提示可能会有帮助:

1) \hat{T} 和 \hat{R} 的定义偏向于概念性而非可操作性。对于有限 MDP, 我们可能用一个大小为 $O(|\mathbf{S}|^2|\mathbf{A}|)$ 的矩阵来表示 \hat{T} , 用一个长度为 $O(|\mathbf{S}||\mathbf{A}|)$ 的向量来表示 \hat{R} 。对于结构化的 MDP, 可以采用更紧凑的表达方式。例如, 线性动力学系统的 MDP 可以用有限维度的矩阵来表示 [Strehl and Littman, 2008b; Brunskill et al, 2009], 而因子状态的 MDP 则可以采用动态贝叶斯网来表示 [Kearns and Koller, 1999]。

2) 没有必要每一步都去更新 \hat{T} 和 \hat{R} 或者重新计算 Q_t , 已知状态-动作对 MDP \hat{M} (以及由之而来的 Q_M^* 和 Q_t) 始终保持不变, 除非某个未知的状态-动作对变成已知的。因此, 我



们仅仅在 A_T 或 A_R 获知新的样本时 (算法 13 中的第 13 行或第 16 行) 才需要更新 \hat{M} 和 Q_i 。

3) 没有必要对所有的 (s, a) 都计算 Q_i , 按照定理 6.1, 要保证 Q_i 在状态 s_i 下是 ε_P 精确的, 只需要确保: $|Q_i(s_i, a) - Q_{\hat{M}}^*(s_i, a)| < \varepsilon_P$ 对于所有的 $a \in \mathbf{A}$ 都成立即可。这样的局部规划相对于全局规划而言, 往往可以大大节省计算量。

4) 给定近似 MDP \hat{M} 和当前状态 s_i , 算法为该状态计算一个近似最优的动作。对于有限 MDP 而言这一步骤可以通过动态规划解决。广义而论, 这样做是非常消耗计算资源的。然而, 幸运的是, 最近在近似局部规划的研究进展表明, 大规模问题使用这样的方法其实也是可行的 [Kearns et al 2002; Kocsis and Szepesvári, 2006; Walsh et al, 2010a]。

KWIK-Rmax 的样本复杂度通过下列定理给出, 它显示算法探索的样本复杂度随 KWIK 学习 MDP 边界的扩大而线性增长。对于每个 KWIK 可学习 MDP 类 \mathbf{M} , KWIK-Rmax 算法自动实例化为一个带有 KWIK 学习器的 \mathbf{M} 的 PAC-MDP 算法。

定理 6.3 令 \mathbf{M} 表示一类 MDP, 它有一个状态空间 \mathbf{S} 和动作空间 \mathbf{A} 。如果 \mathbf{M} 可以由算法 A_T (学习状态转换函数) 以及 A_R (学习回报函数) 高效地通过 KWIK 方式学习, 分别在 KWIK 边界 B_i 和 B_R 下, 那么 KWIK-Rmax 是 \mathbf{M} 的 PAC-MDP。特别的, 如果部分参数如下取值,

$$\varepsilon_T = \frac{\varepsilon(1-\gamma)}{16V_{\max}}, \varepsilon_R = \frac{\varepsilon(1-\gamma)}{16}, \varepsilon_P = \frac{\varepsilon(1-\gamma)}{24}, \delta_T = \delta_R = \frac{\delta}{4}$$

那么, KWIK-Rmax 算法的探索样本复杂度就是

$$O\left(\frac{\max}{\varepsilon(1-\gamma)}\left(B_T(\varepsilon_T, \delta_T) + B_R(\varepsilon_R, \delta_R) + \ln \frac{1}{\delta}\right) \ln \frac{1}{\varepsilon(1-\gamma)}\right)$$

我们省略了 Rmax 的泛化证明 (定理 6.1)。取而代之, 我们描述了几类 KWIK 可学习的 MDP, 并展示 KWIK-Rmax 如何归一化和扩展为前述的 PAC-MDP 算法。由于转换概率函数通常比回报函数更难以学习, 我们仅仅描述如何采用 KWIK 方式学习转换函数。回报函数可以通过同样的算法原理进行 KWIK 学习。更多的例子参见 [Li et al, 2011]。

示例 6.6 显而易见, KWIK-Rmax 是 Rmax 算法的泛化。事实上, Rmax 可以解释为 KWIK-Rmax 采用元算法 (输入分块) 按照 KWIK 方式学习一个有 n 个状态和 m 个动作的有限 MDP。这个元算法记作 A^* , 运行 $m \times n$ 个通过状态-动作对进行索引的骰子学习 (dice-learning) 的实例。每当 A^* 接收到一个输入 $x = (s, a) \in \mathbf{S} \times \mathbf{A}$, 它会根据索引 (s, a) 查询 x , 并做出同样的预测作为该索引下的实例。如果预测是 \perp , A^* 将进行一次取样 (在该情形下, 取样的结果是 (s, a) 发生后的下一个状态), 并将取样的结果传递给相关的骰子学习实例去学习。精确度要求将以较高概率得到满足, 因为所有骰子学习的合理的预测都是高概率 ε 精确的。如果忽略算法因素, A^* 的 KWIK 边界是所有骰子学习算法实例的 KWIK 边界总和, 即 $\tilde{O}(n^2 m / \varepsilon^2)$ 。替换定理 6.3 的边界为该边界, 就变为定理 6.2。

示例 6.7 在很多机器人学或自适应控制应用中, 其操作的系统的状态和动作空间都是有限的, 而且它们的动力学都是线性方程 [Abbeel and Ng, 2005; Strehl and Littman, 2008b], 这里, $\mathbf{S} \subseteq \mathbb{R}^{n_S}$ 和 $\mathbf{A} \subseteq \mathbb{R}^{n_A}$ 分别是状态向量以及动作空间。状态转换遵循一个多元正态分布: 给定当前状态动作对 (s, a) , 下一个状态 s' 从 $\mathcal{N}(F\phi(s, a), \Sigma)$ 中随机抽取, 其中 $\phi: \mathbb{R}^{n_S+n_A} \rightarrow \mathbb{R}^n$ 是一个基函数, 它满足 $\|\phi(\cdot;)\| \leq 1$, $F \in \mathbb{R}^{n_S \times n}$ 是一个矩阵, $\Sigma \in \mathbb{R}^{n_S \times n_S}$ 是一个协方差矩阵。我们假设 ϕ 和 Σ 是给定的, 但 F 是未知的。



对于这样的 MDP, 均值向量 $\mathbf{E}[s']$ 的每一个元件都是关于 $\varphi(s, a)$ 的一个线性函数。因此, n_s 个带噪线性回归的实例可以应用到 KWIK 框架下对 F 中的 n_s 行进行学习^①。如果 $\mathbf{E}[s']$ 的独立元件可以通过子算法精准预测, 我们可以很容易地连接它们用来预测整体的向量 $\mathbf{E}[s']$ 。否则, 我们可以预测 \perp 并获取数据给予算法学习。这种元算法, 称为联合输出 (output combination), 允许我们通过 KWIK 框架学习类似这样的线性参数化的 MDP, 其 KWIK 边界为 $\tilde{O}(n_s^2 n/\epsilon^4)$ 。因此, 根据定理 6.3, 相应的 KWIK-Rmax 实例化是 PAC-MDP。

有一类相关的 MDP, 是由机器人导航任务启发而来的正常偏置模型 [Brunskill et al, 2009], 它可以视作是线性动力学 MDP 的一个特殊的情况。

示例 6.8 因子状态的表达形式是一种特殊类型 MDP 的紧凑表达。令 $m=|\mathbf{A}|$ 为动作数, 每个状态都是一个向量, 它由 n 个元素 $s=(s[1], s[2], \dots, s[n]) \in \mathbf{S}$ 组成。每个元素 $s[i]$ 都称为一个状态变量, 并且从有限集合 \mathbf{S}_i 中取值。整个状态空间 \mathbf{S} 是 $\mathbf{S}_1 \times \dots \times \mathbf{S}_n$ 。假定状态转换函数为 n 个状态转换函数的因子的总乘积, 对于每一个状态变量, 有:

194

$$T(s, a, s') = \prod_{i=1}^n T_i(\mathbf{P}_i(s), a, s'[i])$$

其中 $\mathbf{P}_i(s) \subseteq \{s[1], s[2], \dots, s[n]\}$, 称为第 i 个父集, 它包含与定义 $s'[i]$ 的转换概率有关的状态变量。定义两个量: $D \stackrel{\text{def}}{=} \max_i |\mathbf{P}_i|$, 它表示父集的最大可能的大小; 以及 $N \stackrel{\text{def}}{=} \max_i |\mathbf{S}_i|$, 它定义一个状态变量最多有多少种取值。因此, 尽管因子形式的 MDP 有 mN^{2n} 个转换概率, 但 MDP 实际上只需要 mnN^{D+1} 个自由变量就能确定。当 $D \ll n$ 时, 这样一个 MDP 不仅能得到高效的表达, 还能以一个更低的 KWIK 边界进行 KWIK 学习, 接下来会解释这一点。

当父集 \mathbf{P}_i 是一个先验, 联合输出可以用来联合预测 \mathbf{T}_i , 而且每一个 \mathbf{T}_i 都可以通过一个应用于骰子学习 (对于 N 个元素的多项分布) 的输入分块的实例 (在所有 mN^D 个状态-动作对上) 进行 KWIK 学习。此三阶 KWIK 算法提供了一个方法用来学习因子状态的 MDP 的转换函数, 其 KWIK 边界 [Li et al, 2011] 是: $\tilde{O}(n^3 m D N^{D+1} / \epsilon^2)$ 。这个发现可用于推导因子状态的 MDP 问题的 PAC-MDP 算法 [Guestrin et al, 2002; Kearns and Koller, 1999; Strehl, 2007a]。

在更有趣的情形下, \mathbf{P}_i 是未知的, RL 学习器必须学习因子状态表达的结构。此时, 高效探索问题就变得非常具有挑战性, 因为要同时考虑对 MDP 问题的因子状态表达结构的学习需求 [Kearns and Koller, 1999]。幸运的是, 假设存在知识 D , 我们可以利用带噪联合算法 [Li et al, 2011] 对可能的一系列因子状态表达结构进行 KWIK 学习, 只要 D 比较小, 学习过程不会太耗时。对于上述未知结构情形, 结合带噪联合算法和三阶 KWIK 算法, 不仅能简化一个原有的结构学习算法 [Strehl et al, 2007], 还有助于开发出更多高效的算法 [Diuk et al, 2009]。

上述的示例展示了 KWIK 模型的威力, 当 KWIK 模型和通用 PAC-MDP 结合时, 能够得到定理 6.3。特别是这些示例展示了如何利用 KWIK 模型学习各种不同的重要类型的 MDP, 而其中能立即转变为一个 KWIK-Rmax 的实例的 MDP 都是 PAC-MDP。

我们总结本小节, 有两个重要的开放性问题, 它们是关于 KWIK 模型在设计 PAC-MDP 算法上的使用问题:

① 有一种技术微妙地存在着: 我们需要首先带有有界支持度地夹取一个正态分布把它放入一个分布里, 因为有界的噪声在带噪线性回归中是预先假定好的。



- 当可实现假设被否决时, 即当 $h^* \notin \mathbf{H}$ 时, KWIK 学习一个假设类就变得非常具有挑战性。在这种情形下, 在定义 6.9 中的某种直接前向的精度要求适配器可能会导致 KWIK 模型无法学习一个假设类, 即便这个类其实是可以与其他途径进行 KWIK 学习的 [Li and Littman, 2010]。最近, [Szita and Szepesvári, 2011] 做了一个有趣的研究。
- 当我们谈论 KWIK 可学习 MDP 类时, 一个 MDP M 的维度 $\dim(M)$ 对于不同情形而言都可能不同。大体上来说, 我们无法定义在 KWIK 模型中什么样的假设类复杂度衡量标准是有效的。

6.6 无模型方法

前一小节谈论了在 KWIK 算法的帮助下, 如何系统地为不同类型的 MDP 创建基于模型的 PAC-MDP 强化学习算法。尽管基于模型的算法在实际应用中常常有更好的样本复杂度, 但它们经常需要维护一个未知 MDP 的明确模型, 并且往往需要重复地解决这个近似 MDP (比如在 KWIK-Rmax 中) 以便获取一个价值函数或者采取动作的策略。所以说, 虽然最近在 MDP 的近似规划上有很多进展, 但解决 MDP 问题依然大体上很具有挑战性。

相反, 无模型算法通过一个贪婪策略直接学习最优价值函数, 这是很容易计算的。这些算法常常有较低的平均步骤计算量, 因此也更符合实际应用的实时性要求, 因为在实时性的应用中, 每个动作执行的计算资源是有限的。在有限 MDP 中, 例如, Q 学习和 SARSA 都要求每一步的计算复杂度不能高于 $O(\ln |\mathbf{A}|)$, 因为用到了一个堆结构来维护状态-动作对估计, 而存储这些估计到堆中需要占用的空间复杂度是 $\Theta(|\mathbf{S}||\mathbf{A}|)$ 。相比之下, 求解一个有限 MDP 需要 $\Omega(|\mathbf{S}||\mathbf{A}|)$ 的时间和空间复杂。然而, 这些经典无模型算法没有指定一个明确的探索策略, 但通常采用的技术是 ϵ 贪婪策略 [Sutton and Baro, 1998]。如示例 6.1 所示 (也可参见 [Strehl, 2007b, Section 4.1] 给出的例子), 这些流行的选择却并不像 PAC-MDP 算法那样可靠。存在收敛率往往依赖于诸如覆盖时间这些量 [Even-Dar and Mansour, 2003], 而覆盖时间在探索策略低效的情形下可能是指数级别大的。

延迟 Q 学习 [Strehl et al, 2006b] 是第一个被证明是 PAC-MDP 的无模型算法。它为确定性 MDP [Koenig and Simmons, 1996] 产生早期结果, 并且是某些变体算法与某些随机版本的实时动态规划算法 [Strehl, 2007b] 中很有代表性的一个。该版本算法在算法 14 中给出, 算法 14 对原延迟 Q 算法做了些改进和简化 [Strehl et al, 2006b]: 避免了某个不知为何不自然的探索奖励。

该算法称作“延迟的”是因为它等待一个状态-动作对经历了 m 次后才会更新 Q 值, 其中 m 是一个输入参数。当它更新状态-动作对的 Q 值时, 这个更新可以看成是对最近的 m 个错过更新机会的目标值的平均。从这个角度来看, 延迟 Q 学习的更新规则很像实时动态规划 [Barto et al, 1995], 但它利用样本来逼近贝尔曼算子。

我们注意到, 通过 m 个样本的平均实现的批量更新过程其实和随机梯度下降是等价的, 随机梯度下降以学习率 $1, \frac{1}{2}, \dots, \frac{1}{m}$ 学习这些样本, 假设对 $Q(s, a)$ 的更新并不影响该序列样本。该取样为算法的名字做了解释, 同时强调了 Q 学习和调和关系以及批量更新的本质。

为了鼓励探索, 延迟 Q 学习使用相同的面对不确定性的乐观原则在很多其他算法, 诸如 [Rmax、MBIE and UCRL2]。特别地, 延迟 Q 学习的初始 Q 函数是一个真实函数的过度



估计；在运行过程中，只要 m 足够大，连续的价值函数估计会以很大的概率一直保持过度估计，这一点后面马上就会解释。

```

0: Inputs:  $S, A, \gamma, m \in \mathbb{N}, \xi \in \mathbb{R}_+$ 
1: for all  $(s, a) \in S \times A$  do
2:    $Q(s, a) \leftarrow V_{\max}$  {optimistic initialization of state-action values}
3:    $U(s, a) \leftarrow 0$  {used for attempted updates of the value in  $(s, a)$ }
4:    $B(s, a) \leftarrow 0$  {beginning timestep of attempted update in  $(s, a)$ }
5:    $C(s, a) \leftarrow 0$  {counter for  $(s, a)$ }
6:    $L(s, a) \leftarrow \text{TRUE}$  {the learning flag}
7:    $t^* \leftarrow 0$  {timestamp of most recent state-action value change}
8: for  $t = 1, 2, 3, \dots$  do
9:   Observe the current state  $s_t$ , take action  $a_t \leftarrow \arg \max_{a \in A} Q(s_t, a)$ , receive reward
    $r_t \in [0, 1]$ , and transition to a next state  $s_{t+1}$ .
10:  if  $B(s_t, a_t) \leq t^*$  then
11:     $L(s_t, a_t) \leftarrow \text{TRUE}$ 
12:    if  $L(s_t, a_t) = \text{TRUE}$  then
13:      if  $C(s_t, a_t) = 0$  then
14:         $B(s_t, a_t) \leftarrow t$ 
15:         $C(s_t, a_t) \leftarrow C(s_t, a_t) + 1$ 
16:         $U(s_t, a_t) \leftarrow U(s_t, a_t) + r_t + \gamma \max_{a \in A} Q(s_{t+1}, a)$ 
17:        if  $C(s_t, a_t) = m$  then
18:           $q \leftarrow U(s_t, a_t) / m$ 
19:          if  $Q(s_t, a_t) - q \geq \xi$  {if Q-function changes significantly} then
20:             $Q(s_t, a_t) \leftarrow q$ 
21:             $t^* \leftarrow t$ 
22:          else if  $B(s_t, a_t) > t^*$  then
23:             $L(s_t, a_t) \leftarrow \text{FALSE}$ 
24:             $U(s_t, a_t) \leftarrow 0$ 
25:             $C(s_t, a_t) \leftarrow 0$ 

```

算法 14 延迟 Q 学习

尽管学习标志的角色可能表现得并不明显，但是它们确实能保证在整个延迟 Q 学习算法的执行过程中，延迟更新规则只会发生有限次。当我们为了明确该算法的样本复杂度而证明下列定理的时候，上述事实是很有用的。

定理 6.4 在延迟 Q 学习中，如果参数按照如下方式设置：

$$m = O\left(\frac{V_{\max}^2}{\varepsilon^2(1-\gamma)^2} \ln \frac{|S||A|}{\varepsilon\delta(1-\gamma)}\right) \text{ 且 } \xi = \frac{\varepsilon(1-\gamma)}{8}$$

则该算法是 PAC-MDP 算法，且具有以下样本复杂度：

197

$$O\left(\frac{|S||A|V_{\max}^4}{\varepsilon^4(1-\gamma)^4} \ln \frac{1}{\varepsilon(1-\gamma)} \ln \frac{|S||A|}{\varepsilon\delta(1-\gamma)}\right)$$

值得关注的是，如果状态数量一样，Q 延迟学习算法的样本复杂度和 [Li, 2009] 给出的当前最低边界是一致的。尽管探索的样本复杂度可能比最近提出的 MoRmax 算法 [Szita and Szepesvári, 2010] 要差，但是它的分析中第一次用了“一致状态-动作对”这个有趣的概念，这跟以前的那些基于模型的算法（如 Rmax）不同。令 Q_t 为第 t 时间步长上的价值函数估计，已知状态-动作对集合中的状态-动作对必须满足其值估计有本质上非负的贝尔曼误差：

$$\mathbf{K}_t \stackrel{\text{def}}{=} \left\{ (s, a) \mid R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') \max_{a' \in A} Q_t(s', a') - Q_t(s, a) \geq -\frac{\varepsilon(1-\gamma)}{4} \right\}$$



一些定义可能会在证明中 useful。对每一时间步长，都会对状态-动作对 (s, a) 进行一次尝试更新， $L(s, a) = \text{TRUE}$ ，且 $C(s, a) = m$ ，换句话说，尝试更新的过程按照算法 14 的第 18 ~ 25 行中那样和当前的时间步长相关。另外，如果第 19 行的条件得到满足，尝试更新就是成功的，因为状态-动作对的值已经发生改变了；否则，尝试更新过程是失败的。

证明（粗略的）为了便于论述，我们令 $S = |\mathbf{S}|$ 且 $A = |\mathbf{A}|$ 。我们首先证明，只存在有限次尝试更新发生在算法的整个运行过程中。显然，整个过程中成功的尝试更新的次数最多为： $\kappa \triangleq SAV_{\max}/\xi$ ，因为状态-动作对价值函数初始化为 V_{\max} ，因此价值函数总是非负的，并且每一次成功的尝试更新都会令某个 (s, a) 的 Q 函数值 $Q(s, a)$ 减少至少 ξ 。现在考虑一个固定的状态-动作对 (s, a) 。一旦 (s, a) 出现了 m 次，一次尝试更新就会发生。假设在第 t 时间步长上发生了一次尝试更新。在这之后，下次的尝试更新发生的时间为 $t' > t$ ，这种情况发生时，说明在 t 和 t' 时间段内必定发生了一次成功的尝试更新。我们已经证明至多可能有 κ 次成功更新，所以，最多有 $\kappa + 1$ 次尝试更新。因为存在 SA 个状态-动作对，因此最多可能存在 $SA(1 + \kappa)$ 次尝试更新。这个尝试更新的次数边界可以让我们采用一个联合边界来得到以下结论 [Strehl et al, 2009, Lemma 22]：有较高的概率，每次在 $(s, a) \in \mathbf{K}_t$ 上的尝试更新都会成功，当前必须保证使用确定的 m 值。

我们现在已经准备好了去验证定理 6.1 中提到的那三个条件。利用数学归纳法，乐观条件是最易于验证的。延迟 Q 学习算法使用一个乐观的价值函数的初始化，因此 $Q_t(s, a) \geq Q^*(s, a)$ 并且因此 $V_t(s) \geq V^*(s)$ 对于 $t=1$ 成立。现在，假设 $Q_t(s, a) \geq Q^*(s, a)$ 且 $V_t(s) \geq V^*(s)$ 对所有 (s, a) 都成立。如果某个 $Q(s, a)$ 更新了，那么霍夫丁不等式和有确定值的 m 一起可以确保新的 Q 值依然保持乐观（因为存在近似误差，两者有一个 $O(\epsilon)$ 的差异值）。由于仅存在有限多次尝试更新，我们可以应用联合边界， $V_t(s) \geq V^*(s) - \epsilon/4$ 对于所有 t 都以较高的概率成立。

精确度条件可以利用 \mathbf{K}_t 的定义来验证，根据定义，在已知状态-动作对中贝尔曼误差至少是 $-\epsilon(1-\gamma)/4$ 。另一方面，根据定义 6.7，未知状态-动作对的贝尔曼误差为 0。因此，著名的贝尔曼算子的单调特性暗示 Q_t 是均匀地比已知状态动作 MDP $M_{\mathbf{K}_t}$ 的最优 Q 函数值要大，且两者之间的差异是 $O(\epsilon)$ 。所以精确度条件是成立的。

至于有界变化条件，我们首先观察到 $Q(\cdot, \cdot)$ 更新的次数最多是 κ ，就像上述所言。至于对未知状态-动作对的访问次数，可以说，一次对未知状态-动作对的 Q 值尝试更新操作以较高的概率是成功的（ m 的值必须是固定的），而且一次不成功的更新就会有 $L(s, a) = \text{FALSE}$ ，意味着 $(s, a) \in \mathbf{K}_t$ 。如果想得到一个更为正式的论述则显然需要做更多的工作，但当我们限定了上述尝试更新的次数时，有界变化的正式的论述其实和前面面对学习标志的推理是类似的 [Strehl et al, 2009, Lemma 25]。

延迟 Q 学习可以和间隔估计等技术相结合，以便获得更多效率上的提升 [Strehl, 2007b]。尽管像延迟 Q 学习这样的 PAC-MDP 算法是为了求解有限 MDP 而存在的，但想要扩展这个算法并应用到通用 MDP 问题上却面临很大的挑战，在这一点上和基于模型的 Rmax 算法完全不同。造成这一点的部分原因是分析无模型算法本身是很困难的，因为经常需要通过自举的方式来更新价值函数。在基于模型的算法中，学习目标是固定不变的（目标就是 MDP 模型），但是在无模型的算法中，往往并非如此。在最近的研究工作 [Li and Littman, 2010] 中，通用 MDP 的一个有限域强化学习问题被归约为一组 KWIK 回归问题，所以只要每个独立的 KWIK 回归问题是可解的，那么我们就可以得到一个无模型的 PAC-MDP 强化学习算法。



然而很不幸, 目前为止, 首先怎样设计一个减量问题的基于 KWIK 的无模型强化学习算法都尚无定论, 遑论以后如何将该问题转化为一个有限域强化学习问题了。

6.7 总结

探索是强化学习中的一个根本问题。高效的探索意味着一个强化学习器能够通过在一个未知环境中进行试验而快速得出一个近似最优策略。另一方面, 低效的探索往往导致学习器向近似最优策略收敛的速度缓慢 (或者甚至不收敛)。本章研究在相关文献中被证明高效的探索模式。我们在侧重 PAC-MDP 这个形式化框架的同时, 也简要地概述了其他的替代方案, 包括贝叶斯探索 (Bayesian Exploration) 和遗憾最小化 (Regret Minimization) 等方法, 并且将它们和 PAC-MDP 做了比较。

PAC-MDP 框架为研究探索问题提供了一个严格的框架, 它同时还具有很大的灵活性。通过它, 我们可以同时开发基于模型的和无模型的 PAC-MDP 算法。更重要的是, 在 KWIK 模型的帮助下, 我们可以发展 PAC-MDP 算法使之应用于远不止有限 MDP 的其他更多有用的 MDP 类。本章示例包括了各类问题, 如因子状态 MDP [Guestrin et al, 2002; Kearns and Koller, 1999; Strehl et al, 2007; Diuk et al, 2009]、连续状态 MDP [Strehl and Littman, 2008b; Brunskill et al, 2009] 以及关联 MDP [Walsh, 2010]。此外, KWIK 模型也可以结合学徒制学习, 使得强化学习系统可以在监督者的帮助下更有效地探索 [Walsh et al, 2010b; Sayedi et al, 2011; Walsh et al, 2012]。

199

虽然最坏情形中的样本复杂度界限可能是保守的, 但已证明其原理算法和分析对于指导更实际的探索方案的开发来说是有用的。许多新颖的算法都由之启发而来, 且在实践中表现良好 [Jong and Stone, 2007; Li et al. 2009; Nouri and Littman, 2009, 2010], 所有这些算法在指导探索中都各自引入了“已知”这个概念的各自的表述形式。其中有几个 PAC-MDP 算法在诸如机器人学 [Brunskill et al, 2009] 和计算机游戏 [Diuk et al, 2008] 这样的非平凡应用中表现突出。

致谢。作者对于 John-Langford、Michael Littman、Alex Strehl、Tom Walsh 以及 Eric Wiewiora 等人对 KWIK 模型的发展和许多 PAC-MDP 算法的样本复杂度分析做出的重要贡献表示诚挚的感谢。另外, 作者十分感谢本章的审稿人和编辑以及 Michael Littman 和 Tom Walsh, 感谢他们以各种方式对本文的内容和表述形式提供宝贵的意见。

参考文献

- Abbeel, P., Ng, A.Y.: Exploration and apprenticeship learning in reinforcement learning. In: Proceedings of the Twenty-Second International Conference on Machine Learning (ICML-2005), pp. 1–8 (2005)
- Asmuth, J., Li, L., Littman, M.L., Nouri, A., Wingate, D.: A Bayesian sampling approach to exploration in reinforcement learning. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009), pp. 19–26 (2009)
- Bartlett, P.L., Tewari, A.: REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. In: Proceedings of the Twenty-Fifth Annual Conference on Uncertainty in Artificial Intelligence (UAI-2009), pp. 35–42 (2009)
- Barto, A.G., Bradtke, S.J., Singh, S.P.: Learning to act using real-time dynamic programming. *Artificial Intelligence* 72(1-2), 81–138 (1995)
- Bertsekas, D.P., Tsitsiklis, J.N.: *Neuro-Dynamic Programming*. Athena Scientific (1996)



- Brafman, R.I., Tennenholtz, M.: R-max—a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research* 3, 213–231 (2002)
- Brunskill, E., Leffler, B.R., Li, L., Littman, M.L., Roy, N.: Provably efficient learning with typed parametric models. *Journal of Machine Learning Research* 10, 1955–1988 (2009)
- Burnetas, A.N., Katehakis, M.N.: Optimal adaptive policies for Markov decision processes. *Mathematics of Operations Research* 22(1), 222–255 (1997)
- Dearden, R., Friedman, N., Andre, D.: Model based Bayesian exploration. In: *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-1999)*, pp. 150–159 (1999)
- Diuk, C., Cohen, A., Littman, M.L.: An object-oriented representation for efficient reinforcement learning. In: *Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML-2008)*, pp. 240–247 (2008)
- Diuk, C., Li, L., Leffler, B.R.: The adaptive k -meteorologists problem and its application to structure discovery and feature selection in reinforcement learning. In: *Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML-2009)*, pp. 249–256 (2009)
- Duff, M.O.: Optimal learning: Computational procedures for Bayes-adaptive Markov decision processes. PhD thesis, University of Massachusetts, Amherst, MA (2002)
- Even-Dar, E., Mansour, Y.: Learning rates for Q-learning. *Journal of Machine Learning Research* 5, 1–25 (2003)
- Even-Dar, E., Mannor, S., Mansour, Y.: Multi-Armed Bandit and Markov Decision Processes. In: Kivinen, J., Sloan, R.H. (eds.) *COLT 2002. LNCS (LNAI)*, vol. 2375, pp. 255–270. Springer, Heidelberg (2002)
- Fiechter, C.N.: Efficient reinforcement learning. In: *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory (COLT-1994)*, pp. 88–97 (1994)
- Fiechter, C.N.: Expected mistake bound model for on-line reinforcement learning. In: *Proceedings of the Fourteenth International Conference on Machine Learning (ICML-1997)*, pp. 116–124 (1997)
- Guestrin, C., Patrascu, R., Schuurmans, D.: Algorithm-directed exploration for model-based reinforcement learning in factored MDPs. In: *Proceedings of the Nineteenth International Conference on Machine Learning (ICML-2002)*, pp. 235–242 (2002)
- Jaakkola, T., Jordan, M.I., Singh, S.P.: On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation* 6(6), 1185–1201 (1994)
- Jaksch, T., Ortner, R., Auer, P.: Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research* 11, 1563–1600 (2010)
- Jong, N.K., Stone, P.: Model-based function approximation in reinforcement learning. In: *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2007)*, pp. 670–677 (2007)
- Kaelbling, L.P.: *Learning in Embedded Systems*. MIT Press, Cambridge (1993)
- Kakade, S.: On the sample complexity of reinforcement learning. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, UK (2003)
- Kakade, S., Kearns, M.J., Langford, J.: Exploration in metric state spaces. In: *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, pp. 306–312 (2003)
- Kearns, M.J., Koller, D.: Efficient reinforcement learning in factored MDPs. In: *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-1999)*, pp. 740–747 (1999)
- Kearns, M.J., Singh, S.P.: Finite-sample convergence rates for Q-learning and indirect algorithms. In: *Advances in Neural Information Processing Systems (NIPS-1998)*, vol. 11, pp. 996–1002 (1999)
- Kearns, M.J., Singh, S.P.: Near-optimal reinforcement learning in polynomial time. *Machine Learning* 49(2-3), 209–232 (2002)
- Kearns, M.J., Mansour, Y., Ng, A.Y.: A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning* 49(2-3), 193–208 (2002)



- Kocsis, L., Szepesvári, C.: Bandit Based Monte-Carlo Planning. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 282–293. Springer, Heidelberg (2006)
- Koenig, S., Simmons, R.G.: The effect of representation and knowledge on goal-directed exploration with reinforcement-learning algorithms. *Machine Learning* 22(1-3), 227–250 (1996)
- Kolter, J.Z., Ng, A.Y.: Near Bayesian exploration in polynomial time. In: Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML-2009), pp. 513–520 (2009)
- Li, L.: A unifying framework for computational reinforcement learning theory. PhD thesis, Rutgers University, New Brunswick, NJ (2009)
- Li, L., Littman, M.L.: Reducing reinforcement learning to KWIK online regression. *Annals of Mathematics and Artificial Intelligence* 58(3-4), 217–237 (2010)
- Li, L., Littman, M.L., Mansley, C.R.: Online exploration in least-squares policy iteration. In: Proceedings of the Eighteenth International Conference on Agents and Multiagent Systems (AAMAS-2009), pp. 733–739 (2009)
- Li, L., Littman, M.L., Walsh, T.J., Strehl, A.L.: Knows what it knows: A framework for self-aware learning. *Machine Learning* 82(3), 399–443 (2011)
- Littlestone, N.: Learning quickly when irrelevant attributes abound: A new linear-threshold algorithms. *Machine Learning* 2(4), 285–318 (1987)
- Meuleau, N., Bourguin, P.: Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning* 35(2), 117–154 (1999)
- Moore, A.W., Atkeson, C.G.: Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning* 13(1), 103–130 (1993)
- Neu, G., György, A., Szepesvári, C., Antos, A.: Online Markov decision processes under bandit feedback. In: Advances in Neural Information Processing Systems 23 (NIPS-2010), pp. 1804–1812 (2011)
- Ng, A.Y., Harada, D., Russell, S.J.: Policy invariance under reward transformations: Theory and application to reward shaping. In: Proceedings of the Sixteenth International Conference on Machine Learning (ICML-1999), pp. 278–287 (1999)
- Nouri, A., Littman, M.L.: Multi-resolution exploration in continuous spaces. In: Advances in Neural Information Processing Systems 21 (NIPS-2008), pp. 1209–1216 (2009)
- Nouri, A., Littman, M.L.: Dimension reduction and its application to model-based exploration in continuous spaces. *Machine Learning* 81(1), 85–98 (2010)
- Poupart, P., Vlassis, N., Hoey, J., Regan, K.: An analytic solution to discrete Bayesian reinforcement learning. In: Proceedings of the Twenty-Third International Conference on Machine Learning (ICML-2006), pp. 697–704 (2006)
- Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley-Interscience, New York (1994)
- Ratitch, B., Precup, D.: Using MDP Characteristics to Guide Exploration in Reinforcement Learning. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 313–324. Springer, Heidelberg (2003)
- Robbins, H.: Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society* 58(5), 527–535 (1952)
- Sayedi, A., Zadimoghaddam, M., Blum, A.: Trading off mistakes and don't-know predictions. In: Advances in Neural Information Processing Systems 23 (NIPS-2010), pp. 2092–2100 (2011)
- Singh, S.P., Jaakkola, T., Littman, M.L., Szepesvári, C.: Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning* 38(3), 287–308 (2000)
- Strehl, A.L.: Model-based reinforcement learning in factored-state MDPs. In: Proceedings of the IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning, pp. 103–110 (2007a)
- Strehl, A.L.: Probably approximately correct (PAC) exploration in reinforcement learning. PhD thesis, Rutgers University, New Brunswick, NJ (2007b)
- Strehl, A.L., Littman, M.L.: An analysis of model-based interval estimation for Markov de-



- cision processes. *Journal of Computer and System Sciences* 74(8), 1309–1331 (2008a)
- Strehl, A.L., Littman, M.L.: Online linear regression and its application to model-based reinforcement learning. In: *Advances in Neural Information Processing Systems 20 (NIPS-2007)*, pp. 1417–1424 (2008b)
- Strehl, A.L., Li, L., Littman, M.L.: Incremental model-based learners with formal learning-time guarantees. In: *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence (UAI-2006)*, pp. 485–493 (2006a)
- Strehl, A.L., Li, L., Wiewiora, E., Langford, J., Littman, M.L.: PAC model-free reinforcement learning. In: *Proceedings of the Twenty-Third International Conference on Machine Learning (ICML-2006)*, pp. 881–888 (2006b)
- Strehl, A.L., Diuk, C., Littman, M.L.: Efficient structure learning in factored-state MDPs. In: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI-2007)*, pp. 645–650 (2007)
- Strehl, A.L., Li, L., Littman, M.L.: Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research* 10, 2413–2444 (2009)
- Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press, Cambridge (1998)
- Szita, I., Lőrincz, A.: The many faces of optimism: A unifying approach. In: *Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML-2008)*, pp. 1048–1055 (2008)
- Szita, I., Szepesvári, C.: Model-based reinforcement learning with nearly tight exploration complexity bounds. In: *Proceedings of the Twenty-Seventh International Conference on Machine Learning (ICML-2010)*, pp. 1031–1038 (2010)
- Szita, I., Szepesvári, C.: Agnostic KWIK learning and efficient approximate reinforcement learning. In: *Proceedings of the Twenty-Fourth Annual Conference on Learning Theory, COLT-2011* (2011)
- Tewari, A., Bartlett, P.L.: Optimistic linear programming gives logarithmic regret for irreducible MDPs. In: *Advances in Neural Information Processing Systems 20 (NIPS-2007)*, pp. 1505–1512 (2008)
- Thrun, S.: The role of exploration in learning control. In: White, D.A., Sofge, D.A. (eds.) *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*, pp. 527–559. Van Nostrand Reinhold (1992)
- Valiant, L.G.: A theory of the learnable. *Communications of the ACM* 27(11), 1134–1142 (1984)
- Walsh, T.J.: *Efficient learning of relational models for sequential decision making*. PhD thesis, Rutgers University, New Brunswick, NJ (2010)
- Walsh, T.J., Szita, I., Diuk, C., Littman, M.L.: Exploring compact reinforcement-learning representations with linear regression. In: *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*, pp. 591–598 (2009); corrected version as Technical Report DCS-tr-660, Department of Computer Science, Rutgers University
- Walsh, T.J., Goschin, S., Littman, M.L.: Integrating sample-based planning and model-based reinforcement learning. In: *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-2010)*, pp. 612–617 (2010a)
- Walsh, T.J., Subramanian, K., Littman, M.L., Diuk, C.: Generalizing apprenticeship learning across hypothesis classes. In: *Proceedings of the Twenty-Seventh International Conference on Machine Learning (ICML-2010)*, pp. 1119–1126 (2010b)
- Walsh, T.J., Hewlett, D., Morrison, C.T.: Blending autonomous and apprenticeship learning. In: *Advances in Neural Information Processing Systems 24, NIPS-2011* (2012)
- Watkins, C.J., Dayan, P.: *Q-learning*. *Machine Learning* 8, 279–292 (1992)
- Whitehead, S.D.: Complexity and cooperation in Q-learning. In: *Proceedings of the Eighth International Workshop on Machine Learning (ICML-1991)*, pp. 363–367 (1991)
- Wiering, M., Schmidhuber, J.: Efficient model-based exploration. In: *Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior: From Animals to Animats 5 (SAB-1998)*, pp. 223–228 (1998)

建设性的表征方向

连续状态和动作空间中的强化学习

Hado van Hasselt

摘要

目前,许多传统的强化学习算法可以解决具有小的有限的状态和动作空间中的问题。由于噪声和延迟增强,在这样的离散问题中学习可能是困难的。然而,许多实际的问题具有连续状态或者动作空间,使得学习一个好的决策变得更加复杂。本章讨论如何自动地在连续域中搜索好的决策。因为从连续模型分析计算一个好的决策是不可行的,在本章中,我们将主要集中在讨论显式地更新价值函数、策略或者两者的方法。我们讨论为这些函数选择一个合适的表示的思路,以及基于梯度和无梯度的方法来更新参数。我们将展示如何将这些方法应用到强化学习问题中,并且讨论许多具体的算法。其中,涵盖基于梯度的时序差分学习、进化策略、策略梯度算法和(自然)演员-评论家方法。我们讨论不同方法的优势,并且以实证比较最新的演员-评论家方法和最新的进化策略的性能。

7.1 简介

在本章中,我们考虑连续域中具有延迟回报信号的时序决策问题。完整的问题需要一个算法去学习如何从一个无限大的动作空间中选择动作,从而在一个无限大的状态空间中优化一个噪声延迟累积回报信号,其中,即使是一个单一动作的结果也可以是随机的。这一算法的优良特性包括,在一般性问题的许多不同实例中的适用性和计算效率,因此它可以用在实时和采样效率中,用有限的经验知识学习好的动作选择策略。

由于在连续空间中完整的强化学习问题的复杂性,已有许多传统的强化学习方法用来解决具有小的有限状态和动作空间的马尔可夫决策过程(Markov Decision Process, MDP)。然而,许多问题本身具有大的或者连续的域。在本章中,我们讨论怎样使用强化学习在具有连续状态空间和离散动作空间的MDP中,以及在状态和动作空间都是连续的MDP中学习好的动作选择策略。

在本章中,我们假设环境模型是未知的。如果能够找到一个模型,我们就可以使用动态规划[Bellman, 1957; Howard, 1960; Puterman, 1994; Sutton and Barto, 1998; Bertsekas, 2005, 2007]或者从模型中采样并且使用以下讨论的一种强化学习算法。我们主要集中在控制问题上,这意味着我们想要找到产生高回报的动作选择策略,与预测问题相反,预测用来估计已知策略的值。

为了从不同的角度对强化学习进行一般性介绍,我们参考了以下书籍[Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998],以及最新的书籍[Bertsekas, 2007; Powell, 2007; Szepesvári, 2010; Busoniu et al, 2010]。每当我们提到一个章节,表明该章是来自同一卷的相关章节。

在本章节的剩余部分，我们描述连续域中的 MDP 的结构，并且讨论在连续域的 MDP 中找到好的策略的三个通用的方法论。在 7.2 节中，我们讨论用来处理大的或者连续空间的函数估计技巧。在 7.3 节中，我们将这些技巧应用到强化学习中，讨论连续域中强化学习的当前知识状态，包括时序差分、策略梯度、演员 – 评论家算法和进化算法的讨论。7.4 节展示了实验结果，在一个双极车杆问题中比较了演员 – 评论家算法和进化算法。7.5 节总结了本章内容。

7.1.1 连续域中的马尔可夫决策过程

一个马尔可夫决策过程是一个 5 元组 (S, A, T, R, γ) 。在本章中，状态空间 S 通常是一个无限大的有界集合。更具体地说，我们假设状态空间是一个可能的多维欧几里得空间的一个子集，因此， $S \subseteq \mathbb{R}^{D_s}$ ，其中， $D_s \in \mathbb{N}$ 是状态空间的维度。动作空间是离散的或者连续的，在后者中，我们假设 $A \subseteq \mathbb{R}^{D_a}$ ，其中 $D_a \in \mathbb{N}$ 是动作空间的维度[⊖]。我们考虑两个变体：具有连续状态和离散动作的 MDP 以及状态和动作都是连续的 MDP。一般来说，当我们写“连续的”时，结果与写“大且有限”一致。本章中用的符号见表 7.1。

表 7.1 本章中使用的符号。所有的向量为列向量

$D_x \in \{1, 2, \dots\}$	空间 X 的维度
$S \subseteq \mathbb{R}^{D_s}$	状态空间
$A \subseteq \mathbb{R}^{D_a}$	动作空间
$T: S \times A \times S \rightarrow [0, 1]$	状态转换函数
$R: S \times A \times S \rightarrow \mathbb{R}$	期望的回报函数
$\gamma \in [0, 1]$	阻尼系数
$V: S \rightarrow \mathbb{R}$	状态价值函数
$Q: S \times A \rightarrow \mathbb{R}$	状态 – 动作价值函数
$\pi: S \times A \rightarrow [0, 1]$	动作选择策略
$\alpha \in \mathbb{R}, \beta \in \mathbb{R}$	步长参数（可能依赖于状态和动作）
$t \in \mathbb{N}$	时间步长
$k \in \mathbb{N}$	片段
$\Phi \subseteq \mathbb{R}^{D_\phi}$	特征空间
$\phi: S \rightarrow \Phi$	特征提取函数
$\Theta \subseteq \mathbb{R}^{D_\theta}$	价值函数的参数空间
$\theta \in \Theta$	一个价值函数的参数向量
$\Psi \subseteq \mathbb{R}^{D_\psi}$	策略的参数空间
$\psi \in \Psi$	一个策略的参数向量
$\mathbf{e} \in \mathbb{R}^{D_e}$	资格迹向量
$\ \mathbf{x}\ = \sum_{i=0}^n x[i]^2$	向量 $\mathbf{x} = \{x[0], \dots, x[n]\}$ 的二范数
$\ f\ = \int_{x \in X} (f(x))^2 dx$	函数 $f: X \rightarrow \mathbb{R}$ 的二范数
$\ f\ _w = \int_{x \in X} w(x)(f(x))^2 dx$	函数 $f: X \rightarrow \mathbb{R}$ 的加权二范数

转换函数 $T(s, a, s')$ 给出了动作 a 在状态 s 下转换为 s' 的概率。当状态空间是连续的，我们假设转换函数指定了一个概率密度函数（PDF），使得

⊖ 总体来说，把状态映射到连续集的映射函数能够更准确地表达一个动作空间，比如 $A(s) \subseteq \mathbb{R}^{D_a}$ 。为了简便起见，我们忽略了这一细微的差别。

$$\int_S T(s, a, s') ds' = P(s_t + 1 \in S' | s_t = s \text{ and } a_t = a)$$

表示动作 a 由状态 s 转换为 $S' \subseteq S$ 中的一个状态的概率。通过一个描述系统动力学的函数来描述这个转换, 这样往往更直观:

$$s_{t+1} = T(s_t, a_t) + \omega_T(s_t, a_t)$$

其中, $T: S \times A \rightarrow S$ 是一个确定的转换函数, 对于给定的状态-动作对返回期望的下一个状态, $\omega_T(s, a)$ 是一个与状态向量大小相同的零均值噪声向量。例如, s_{t+1} 可以从以 $T(s_t, a_t)$ 为中心的高斯分布中采样。回报函数对于任意两个状态和一个动作给出期望回报。实际的回报可以包含噪声:

[209]

$$r_{t+1} = R(s_t, a_t, s_{t+1}) + \omega_R(s_t, a_t, s_{t+1})$$

其中, $\omega_R(s, a, s')$ 是实数值的零均值噪声项。如果 ω_R 以及 ω_T 的成分在所有的时间步长中不是均为零, 那么, MDP 是随机的, 否则, 它是确定的。如果 T 或者 R 是时间依赖的, MDP 是非稳定的。在本章中, 我们假设 MDP 是稳定的。由于我们一般假设 S 、 A 和 γ 是未知的, 因此在本章中当提及模型时, 通常意味着 T 和 R 的近似。

当只有状态空间是连续的, 动作选择策略使用一个状态依赖概率质量函数 $\pi: S \times A \rightarrow [0, 1]$ 表示, 使得

$$\pi(s, a) = P(a_t = a | s_t = s) \text{ 并且 } \sum_{a \in A} \pi(s, a) = 1$$

当动作空间也是连续的时, $\pi(s)$ 表示动作空间的一个概率密度函数。

预测的目标是找到一个给定策略的预期未来折扣回报的值。控制的目标是通过找到最优策略优化这个值。有必要定义下面的算子 $B^\pi: \mathcal{V} \rightarrow \mathcal{V}$, $B^*: \mathcal{V} \rightarrow \mathcal{V}$, 其中, \mathcal{V} 是所有价值函数的空间^①:

$$\begin{aligned} (B^\pi V)(s) &= \int_A \pi(s, a) \int_S T(s, a, s') (R(s, a, s') + \gamma V(s')) ds' da \\ (B^* V)(s) &= \max_a \int_S T(s, a, s') (R(s, a, s') + \gamma V(s')) ds' \end{aligned} \quad (7.1)$$

在连续的 MDP 中, 给定策略的值以及最优值可以用贝尔曼方程表示为 $V^\pi = B^\pi V^\pi$ 以及 $V^* = B^* V^*$ 。 $V^\pi(s)$ 是从状态 s 开始执行策略 π 的值, $V^*(s) = \max_\pi V^\pi(s)$ 是最好的可能策略的值。如果动作空间是有限的, 公式 (7.1) 中的外积分应当被求和代替。在本章中, 我们主要考虑折扣的 MDP, 意味着 $\gamma \in (0, 1)$ 。

对于有限动作空间的控制, 通常使用动作值。连续状态空间的最优行为值由贝尔曼方程给出:

$$Q^*(s, a) = \int_S T(s, a, s') \left(R(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right) ds' \quad (7.2)$$

它的思想是, 当 Q 以充分的精度逼近 Q^* 时, 我们通过在每个状态 S 选择使 $Q(s, a)$ 最大化的参数 a 来获得一个好的策略。不幸的是, 当动作空间是连续的, 公式 (7.2) 中的选择和最大算子可能要求找到一个非平凡最优问题的解。我们在 7.3 节讨论处理连续动作的算法。首先, 我们讨论三个在连续 MDP 中学习好的策略的一般方法。

[210]

7.1.2 求解连续 MDP 的方法

在控制问题中, 目标是最优策略的一个近似。最优策略依赖于最优值, 反过来, 其依赖

① 在文献中, 这些算子通常表示为 T^π 和 T^* (例如 [Szepesvári, 2010]), 但是由于我们用 T 表示转换函数, 所以这里选择使用 B 来表示。

于 MDP 模型。在公式 (7.2) 中, 最优策略是对每一个状态 $\sum_a \pi^*(s, a) Q^*(s, a) = \max_a Q^*(s, a)$ 最大化 Q^* 的策略 π^* 。这意味着, 与其去直接估计 π^* , 不如去估计 Q^* , 或者必要时我们甚至可以估计 T 和 R 来构建 Q^* 和 π^* 。这些观察导致以下三个一般的方法, 它们的不同在于在方案的哪一部分被显式地逼近。这些方法不是相互排斥的, 我们将讨论使用这些方法的组合的算法。

模型逼近。模型逼近算法逼近 MDP, 并且利用该近似 MDP 计算期望的策略。由于假设 S, A 和 γ 是已知的, 这相当于去学习函数 T 和 R 的近似^①。由于马尔可夫的属性, 这些函数只依赖于局部数据。估计这些函数的问题转换成了一个相当标准的监督学习问题。例如, 可以使用贝叶斯方法 [Dearden et al, 1998, 1999; Strens, 2000; Poupart et al, 2006] 来估计要求的模型。学习这个模型可能不简单, 但是, 一般情况下它比学习策略的值或者直接优化策略更容易。关于模型学习算法的最新综述参见 [Nguyen-Tuong and Peters, 2011]。

近似模型可以用来计算价值函数, 这可以迭代地完成, 例如, 使用值迭代或者策略迭代 [Bellman, 1957; Howard, 1960; Puterman and Shin, 1978; Puterman, 1994]。在连续状态 MDP 中, 基于模型的算法的主要缺点是即使一个模型是已知的, 一般来说, 对于所有可能的状态, 也不能容易地从模型中提取一个好的策略。例如, 值迭代在整个状态空间使用一个内循环, 如果空间无限大则这是不可能的。另外, 一个学习到的模型可以用来生成样本运行。这些样本可以用来估计价值函数, 或者使用下面列出的方法之一来改进策略。然而, 如果模型的准确性是值得商榷的, 由此产生的策略可能不比直接基于用来构造近似模型的样本的策略更好。在一些情况下, 值迭代可能是灵活的, 例如, 由于 $T(s, a, s')$ 只对小部分状态 s' 是非零的。即使如此, 直接逼近这个数值可能比从一个近似模型推断出该值更容易。由于篇幅的原因, 我们将不进一步讨论模型逼近。

数值逼近。在这第二个方法中, 样本用来直接逼近 V^* 和 Q^* 。许多强化学习算法属于这一类。我们在 7.3.1 节中讨论数值逼近算法。

211

策略逼近。数值逼近算法通过估计可推断出策略的状态或者动作的价值来直接参数化策略。策略逼近算法直接存储策略并尝试更新策略来逼近最优策略。只存储一个策略而不是一个价值函数的算法, 通常称为直接策略搜索 [Ng et al, 1999] 或者 actor-only 算法 [Konda and Tsitsiklis, 2003]。存储策略和价值函数的算法通常称为 actor-critic 方法 [Barto et al, 1983; Sutton, 1984; Sutton and Barto, 1998; Konda, 2002; Konda and Tsitsiklis, 2003]。我们将讨论这两种方法的例子。使用这一术语, 基于价值的、不存储明确的策略的算法可以认为是 critic-only 算法。策略逼近算法在 7.3.2 节讨论。

7.2 函数逼近

在讨论更新价值函数的近似值或策略的算法之前, 我们讨论一般的存储和更新近似函数的方法。从数据学习函数的一般方法是机器学习领域的研究热点。一般性的讨论参见 [Vapnik, 1995; Mitchell, 1996; Bishop, 2006]。

在 7.2.1 节和 7.2.2 节, 我们讨论了线性和非线性函数逼近。在这两种情况下, 由一组可调参数确定近似函数的值。在 7.2.3 节中, 我们讨论基于梯度和无梯度的方法来更新这些参

① 在工程学中, 回报函数通常是已知的。不幸的是, 这并不能使事情变得更容易, 因为转换函数通常很难估计。

数。这两种方法经常被用在强化学习中,并取得相当大的成功 [Sutton, 1988; Werbos, 1989b, a, 1990; Whitley et al, 1993; Tesauro, 1994, 1995; Moriarty and Miikkulainen, 1996; Moriarty et al, 1999; Whitson and Stone, 2006; Wierstra et al, 2008; Rückstieß et al, 2010]。由于篇幅的限制,我们不讨论非参数的方法,例如基于内核的方法(参见 [Ormonet and Sen, 2002; Powell, 2007; Buşoniu et al, 2010])。

在本节中,我们主要集中在逼近器的一般函数形式和更新参数的一般方法。为了将这些方法应用于强化学习,有大量的设计注意事项。例如,我们必须决定如何衡量近似的准确性。在 7.3 节中我们讨论如何将这方法应用于强化学习。

在监督学习中,给定的标记数据集包含一些输入以及这些输入的预期输出。然后可以回答关于生成数据的过程的统计问题,例如,对于不可见的输入数据生成的函数的值。在基于价值的强化学习中,目标可能取决于适应策略或者适应状态值。因此,在训练过程中,目标可能会改变,并不是监督学习中所有方法都可直接应用于强化学习。尽管如此,许多相同的技术可以成功地用于强化学习设置中,只要关注于 MDP 中固有属性的学习。首先,我们讨论逼近器的选择问题。讨论将分成两部分:线性函数逼近和非线性函数逼近。

7.2.1 线性函数逼近

假设给定某一特征提取函数 $\phi: S \rightarrow \Phi$, 它将状态映射到特征空间 Φ 中的特征。假设 $\Phi \subseteq \mathbb{R}^{D_\phi}$, 其中 D_ϕ 是特征空间的维度。关于选择好的特征的讨论不在本章的范围内,但是该主题可以参考 [Buşoniu et al, 2010]。

线性函数是一个依赖于特征向量的简单的参数函数。例如,一个数值逼近算法,其中,价值函数通过以下方式逼近:

$$V_t(s) = \theta_t^\top \phi(s) \quad (7.3)$$

在公式 (7.3) 和本章余下的部分中, $\theta_t \in \Theta$ 表示在时刻 t 的可适应参数向量, $\phi(s) \in \Phi$ 是状态 s 的特征向量。由于公式 (7.3) 中的函数关于参数是线性的,我们称其为线性函数逼近器。请注意,它关于状态变量可能是非线性的,这依赖于特征提取。在本节中,参数空间的维度 D_θ 等于特征空间 D_ϕ 的维度。这一条件在其他类型的函数逼近中不一定要成立。

线性函数逼近器是有用的,因为它们比非线性函数逼近器更容易理解,将其用于强化学习中,在各种附加的假设下 [Sutton, 1984, 1988; Dayan, 1992; Peng, 1993; Dayan and Sejnowski, 1994; Bertsekas and Tsitsiklis, 1996; Tsitsiklis and Van Roy, 1997], 促成了许多收敛性保证。从实用的角度来看,线性逼近器是有用的,因为它们易于实现并且计算快速。

许多问题具有大的状态空间,每个状态可以用一个有限大小的特征向量来有效地表示。例如,本章后面将考虑的双极车杆问题具有连续的状态变量,因此,具有一个无限大的状态空间。然而,每个状态可以用一个 6 元素的向量表示。这意味着,我们可能需要一个无限大的表,但是,如果使用公式 (7.3),以状态变量作为特征,只需 6 元的参数向量即可。

价值函数的可调参数的减少是有开销的。很显然,并非所有可能的价值函数都可以表示为问题特征的线性组合。因此,我们的解决方案仅限于可以用所选择的函数形式表示的价值函数的集合。如果事先不知道哪些特征对于给定的问题是有用的,则使用非线性函数逼近是有好处的,我们将在 7.2.2 节讨论。

7.2.1.1 离散化状态空间: tile 编码

为线性函数逼近器寻找特征的一个常用的方法是划分连续的状态空间为独立的片段,

将每一个特征依附到每一个片段上。如果相关的状态属于对应的片段，那么特征是有有效的（即，等于1），不然，是无效的（即，等于0）。

这种离散化方法经常被用在强化学习中的例子是 tile 编码 [Watkins, 1989; Lin and Kim, 1991; Sutton, 1996; Santamaria et al, 1997; Sutton and Barto, 1998]。tile 编码基于 [Albus, 1971] 提出的小脑模型关节控制器 (Cerebellar Model Articulation Controller, CMAC) 结构。在 tile 编码中，状态空间被划分成许多不相交的集合。这些集合在本文中通常叫作 tile。例如，可以定义 N 个超立方体，每一个立方体 H_n 通过一个笛卡儿积 $H_n = [x_{n,1}, y_{n,1}] \times \cdots \times [x_{n,D_s}, y_{n,D_s}]$ 定义，其中， $x_{n,d}$ 是超立方体 H_n 关于状态维度 d 的下界， $y_{n,d}$ 是相应的上界。然后，当 $S \in H_n$ 时，与 H_n 相应的特征 $\phi_n(s) \in \phi(s)$ 等于 1，否则，等于 0。

tile 编码背后的想法是使用多个非重叠划分法。如果单个划分包含 N 个 tile，可以利用 M 种这样的划分法获得一个维度为 $D_\phi = MN$ 的特征向量。在每一个状态，这些特征中恰好有 M 个特征等于 1，其他的等于 0。如图 7.1a 所示，划分方法数目 $M=2$ ，特征数目 $D_\phi=24$ 。这些划分不必是齐次的。图 7.1b 展示了一个非齐次的例子，其中 $M=2$ ， $D_\phi=13$ 。

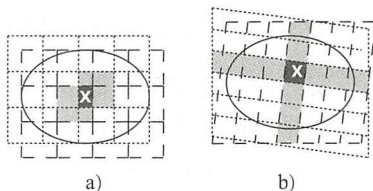


图 7.1 一个椭圆形的状态空间被离散化为两个 tile。对于位于 X 的状态，两个活动的 tile 用淡灰色显示。这些活动特征的叠加显示为深灰色。a) 每个 tile 包含 12 个小 tile。这些特征向量包含 24 个元素，在椭圆状态空间中可以遇到 35 种不同的活动特征组合。b) 特征向量包含 13 个元素和 34 个活动特征的组合，虽然一些组合对应于椭圆的很小的部分

当对于每个状态有 M 个特征是有有效的，多达 $\binom{D_\phi}{M}$ 种不同的情况可以理论上用 D_ϕ 个特征来表示。这与朴素的每个状态只有一个特征有效的方法形成鲜明对比，后者只能表示 D_ϕ 种不同的具有相同特征数量的情况[⊖]。实际上， $\binom{D_\phi}{M}$ 的上界几乎不可获取，这是因为许多有效特征的组合是不合理的。在图 7.1 的两个示例中，不同的特征向量的数目确实大于特征向量的长度并且小于理论上界： $24 < 35 < \binom{24}{2} = 276$ 并且 $13 < 34 < \binom{13}{2} = 78$ 。

7.2.1.2 离散化问题

离散化方法（例如，tile 编码）的一个潜在问题是，所得到的从状态映射到特征空间的函数不是单射的。换句话说， $\phi(s) = \phi(s')$ 并不能推出 $s = s'$ 。这意味着产生的特征空间 MDP 是部分可观察的，应该考虑使用一个显示设计的算法解决部分可观察的 MDP (POMDP)。关于 POMDP 的更多内容参见第 12 章。在实践中，使用 tile 编码已经取得了很多良好的结果，但是，离散化和产生的马尔可夫性质损失意味着，大部分关于普通的强化学习算法的收敛性

⊖ 注意： $1 < M < D_\phi$ 表明 $D_\phi < \binom{D_\phi}{M}$ 。

证明不适用于离散状态空间。这对任何使用不是一个马尔可夫状态空间的单射函数的特征空间的函数逼近都是成立的。

直观地说，这一点可以用几个简单的例子来解释。考虑一个状态空间 $S=\mathbb{R}$ ，进行如下离散化，当 $s \leq -2$ 时，有 $\phi(s)=(1,0,0)^T$ ，当 $-2 < s < 2$ 时， $\phi(s)=(0,1,0)^T$ ，当 $s \geq 2$ 时， $\phi(s)=(0,0,1)^T$ ，动作空间是 $A=\{-2,2\}$ ，转换函数是 $s_{t+1}=s_t+a_t$ ，初始状态是 $s_0=1$ 。回报函数定义如下：当 $s_t \in (-2,2)$ ， $r_{t+1}=1$ ；否则， $r_{t+1}=-1$ 。回报函数和特征映射函数如图 7.2 所示。在这个 MDP 中，在状态 $s=1$ 和 $s=-1$ 间跳动是最优的。然而，如果我们观察特征向量 $(0,1,0)^T$ ，不能知道是否在状态 $s=-1$ 或者 $s=1$ ，并且我们不能确定最优行为。

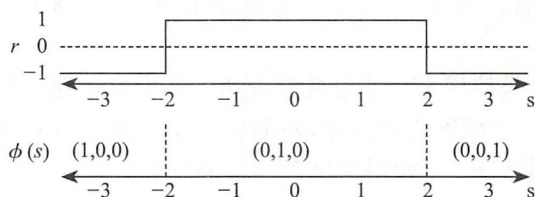


图 7.2 一个回报函数和特征映射。回报是马尔可夫的特征。当 $a_t \in \{-2,2\}$ 时，如果 $s_{t+1}=s_t+a_t$ ，则该特征转换函数不是马尔可夫的。这使得不可能确定最优策略

像 tile 编码这些方法的另一个实际的问题关系到很多算法都使用的步长参数。例如，在 [215] 很多算法中，线性函数逼近器的参数采用以下类似方式进行更新

$$\theta_{t+1} = \theta_t + \alpha_t(s_i) \delta_t \phi(s_i) \quad (7.4)$$

其中 $\alpha_t(s_i) \in [0,1]$ 是步长大小， δ_t 是当前状态值的误差。它可能是一个时序差分误差、当前状态与蒙特卡罗采样的差，或者任何其他相关的误差。这种更新方式的一种推导和解释以及其变体在 7.2.3.1 节以及 7.3.1.2 节中给出。

如果根据公式 (7.4) 观察对一个值 $V(s)=\theta^T \phi(s)$ 的更新，得到

$$\begin{aligned} V_{t+1}(s) &= \theta_{t+1}^T \phi(s) = (\theta_t + \alpha_t(s) \delta_t \phi(s))^T \phi(s) \\ &= \theta_t^T \phi(s) + \alpha_t(s) \phi^T(s) \phi(s) \delta_t \\ &= V_t(s) + \alpha_t(s) \phi^T(s) \phi(s) \delta_t \end{aligned}$$

换句话说，对于这些值的有效的步长大小等于

$$\alpha_t(s_i) \phi^T(s_i) \phi(s_i) = \alpha_t(s_i) \|\phi(s_i)\|^2 \quad (7.5)$$

例如，在 tile 编码中， $\|\phi(s_i)\|^2$ 等于划分 M 的数目。因此，对于 $\alpha_t(s_i) > 1/M$ ，关于价值函数的有效步长大于 1。它可能引起参数的发散。相反地，如果特征向量的欧几里得范数 $\|\phi(s)\|$ 总是小的，价值函数的改变可能比预期更小。

该问题可能在任何特征空间和线性函数逼近中，因此使用公式 (7.5) 中的有效步长来更新价值函数。这表明通过以下方式，适度地放缩步长可能是一个好的想法，

$$\tilde{\alpha}_t(s_i) = \alpha_t(s_i) / \|\phi(s_i)\|^2$$

其中 $\tilde{\alpha}_t(s_i)$ 是缩放后的步长[⊖]。缩放后的步长可以防止对于值的非预期的小或大的更新。

总的来说，对于所有的 s ，确保 $|\phi(s)| = \sum_k \phi_k(s) \leq 1$ 总是一个好主意。例如，在 tile 编码中，可能设置动作特征值为 $1/M$ ，而不是 1。这样的特征表示有很好的收敛特性，因

⊖ 如果 $\|\phi(s_i)\|=0$ 我们可以安全地定义 $\tilde{\alpha}_t(s_i)=0$ ，因为在那种情况下，更新公式 (7.4) 无论如何也不会改变参数。

为它们是非扩张性的,意思是说,对任何特征向量 $\phi(s)$ 和任意两个参数向量 θ 和 θ' , 满足 $\max_k |\phi(s)^\top \theta - \phi(s)^\top \theta'| \leq \max_k |\theta_k - \theta'_k|$ 。一个非扩张性的函数使得证明一个算法通过所谓的收缩映射 [Gordon, 1995; Littman and Szepesvári, 1996; Bertsekas and Tsitsiklis, 1996; Bertsekas, 2007; Szepesvári, 2010; Buşoniu et al, 2010] 按照期望迭代地优化解变得更加简单。实现收缩映射的算法最终达到一个最优的解,并且保证不发散,例如,通过更新它们的参数到无限高的值。

[216]

离散化的一个最终的问题是在函数中引入了不连续性。如果输入改变很小,并且两个输入落在输入空间的不同段,则近似值可能会变化相当大的量。

7.2.1.3 模糊表示

一些离散化的问题可以通过使用一个分段线性函数而不是分段常量函数来避免。一种方式是使用所谓的模糊集 [Zadeh, 1965; Klir and Yuan, 1995; Babuska, 1998]。模糊集是正规集到模糊隶属度的泛化。意思是说,元素可以部分地属于一个集合,而不是完全属于或不属于。

模糊集的一个常见的例子是将温度划分为“冷”和“暖”。在寒冷和温暖之间有一个逐渐的过渡,所以通常说一个特定的温度是部分寒冷和部分温暖是更加自然的。

在强化学习中,状态或状态-动作空间可以划分成模糊集。从而,一个状态可以部分地属于用特征 ϕ_i 定义的集合,部分地属于用特征 ϕ_j 定义的集合。例如,可以有 $\phi_i(s)=0.1$ 和 $\phi_j(s)=0.3$ 。这种观点的一个优势是假设 $\sum_k \phi_k(s) \leq 1$ 是非常自然的,因为一个元素的每个部分可以只属于一个集合。例如,某个物体不可能同时是完全温暖或者完全寒冷的。

定义一个集合,使得特征激活的每个组合精确对应一个状态是可能的,从而避免了之前草拟的局部可观测性问题。一个常见的选择是使用三角函数,在对应特征的中心,三角函数值等于1,进一步地,关于状态从中心线性地衰减为0。注意,可以构建一些函数,使得它们跨越整个状态空间,并且对于所有的状态 $\sum_k \phi_k(s) \leq 1$ 。

模糊强化学习的完整处理超出了本章的范围。使模糊逻辑与强化学习之间明确联系的参考文献包括 [Berenji and Khedkar, 1992; Berenji, 1994; Lin and Lee, 1994; Glorennec, 1994; Bonarini, 1996; Jouffe, 1998; Zhou and Meng, 2003; Buşoniu et al, 2008, 2010]。模糊集的一个缺点是这些集合仍然需要事先定义,这可能是困难的。

7.2.2 非线性函数逼近

线性函数逼近相比于非线性函数逼近的主要缺点是需要好的信息特征^①。经常假设特征是预先挑选的,这可能需要领域知识。即使保证了在极限条件下收敛到一个最优解,这个解也只是在关于给定特征的最可能的线性函数的意义上最优。此外,虽然几乎不能给出理论保证,很好的实证结果已经通过结合强化学习算法和非线性函数逼近器获得了,例如神经网络 [Haykin, 1994; Bishop, 1995, 2006; Ripley, 2008]。实例包括五子棋 [Tesauro, 1992, 1994, 1995]、机器人技术 [Anderson, 1989; Lin, 1993; Touzet, 1997; Coulom, 2002] 和电梯调度 [Crites and Barto, 1996, 1998]。

[217]

在一个参数化非线性函数逼近器中,需要优化的函数用一些预定的参数化函数表示。例如,对于基于值的算法,有

$$V_i(s) = V(\phi(s), \theta_i) \quad (7.6)$$

① 非参数方法在一定程度上减轻了这一需求,但一般来说很难分析。对这些方法的讨论不属于本章的范围。

在这里, $\theta_t \in \Theta$ 的大小没有必要等于 $\phi(s) \in \Phi$ 的大小。例如, V 可能是一个神经网络, 其中, θ_t 是时刻 t 的所有加权值的向量。通常, 函数 V 形式是固定的。然而, 在学习过程中改变函数的结构也是可能的, 例如 [Stanley and Miikkulainen, 2002; Taylor et al, 2006; Whiteson and Stone, 2006; Busoniu et al, 2010]。

总之, 使用相同的输入特征, 与线性逼近器相比, 非线性函数逼近器可能逼近一个具有更好精确度的未知函数。在一些情况下, 通过使用状态变量作为输入, 完全避免定义特征也是可能的。在强化学习中, 非线性函数逼近的一个缺点是只能给出很小的收敛保证。在一些情况下, 可以确保收敛到一个局部最优 (例如 [Maei et al, 2009]), 但是, 总体来说, 相比线性函数逼近, 非线性函数逼近的理论有待进一步发展。

7.2.3 更新参数

一些算法允许封闭形式的参数计算, 对于一组给定的经验样本, 能最好地逼近期望的函数。例如, 当 TD 学习耦合线性函数逼近时, 最小二乘时序差分学习 (LSTD) [Bradtke and Barto, 1996; Boyan, 2002; Geramifard et al, 2006] 可以用来计算参数, 使得在所观察到的转换上的实验时序差分误差最小化。然而, 对于非线性算法 (如 Q 学习) 或者当使用非线性函数逼近时, 这些方法不适用, 并且参数应以不同的方式优化。

下面, 我们解释如何使用梯度下降和无梯度优化这两种通用的技术来调整逼近的参数。这些程序可以用在线性和非线性逼近中, 并且它们都可用于以下三种函数类型: 模型、价值函数和策略。在 7.3 节中, 我们讨论使用这些方法的强化学习算法。

我们将不详细讨论贝叶斯方法, 但是, 这种方法可以用于学习平稳函数的概率分布, 例如, 稳定的 MDP 的回报和转换函数。它的优势是利用在线算法可以选择动作来增加高度不确定性的模型的部分知识。贝叶斯方法在某种程度上不太适用于学习不稳定函数的值, 例如, 变化的策略的值。更多的关于贝叶斯推理的信息详见 [Bishop, 2006 年]。关于在强化学习情境中的贝叶斯方法, 参见 [Dearden et al 1998, 1999; Strens, 2000; Poupart et al 2006] 和第 11 章。

7.2.3.1 梯度下降

梯度下降的更新沿着我们想要最小化的某个参数化函数的负梯度方向。根据一阶泰勒展开, 参数化函数的梯度是指向函数增长方向的参数空间的一个向量。更简单地说, 如果该函数是光滑的, 在梯度方向少量地改变参数值时, 我们期望函数值能略有增加。

在负梯度指向的方向, 函数值预期下降, 所以在这个方向移动参数应该使函数生成更低的值。算法 15 展示了该基本算法。为了简化, 考虑一个实值参数化函数 $\mathbf{E} : \mathbb{R}^N \times \mathbb{R}^P \rightarrow \mathbb{R}$ 。目标是该函数的输出尽可能小。为了达到这个目标, $\theta \in \mathbb{R}^{D_\theta}$ 的参数沿着负梯度方向更新。梯度 $\nabla_\theta \mathbf{E}(x, \theta)$ 是一个列向量, 该列向量是在输入 x 上由 \mathbf{E} 关于参数向量 θ 中元素求得的导数。因为梯度只描述了函数的局部形状, 该算法可能最终在局部达到最小值。

通常, \mathbf{E} 是一个误差的度量, 如时序差分或者预测误差。例如, 考虑一个参数化的逼近回报函数 $\bar{R} : S \times A \times \mathbb{R}^P \rightarrow \mathbb{R}$ 和一个样本 (s_t, a_t, r_{t+1}) 。然后, 我们可以使用 $\mathbf{E}(s_t, a_t, \theta_t) = (\bar{R}(s_t, a_t, \theta_t) - r_{t+1})^2$ 。

如果在超过一个输入-输出对上同时计算梯度, 结果是下面的批量更新

$$\theta_{t+1} = \theta_t - \alpha_t \sum_i \nabla_\theta \mathbf{E}_i(x_i, \theta_t)$$

1: **input:** differentiable function $\mathbf{E} : \mathbb{R}^N \times \mathbb{R}^P \rightarrow \mathbb{R}$ to be minimized,
 step size sequence $\alpha_t \in [0, 1]$, initial parameters $\theta_0 \in \mathbb{R}^P$
 2: **output:** a parameter vector θ such that \mathbf{E} is small
 3: **for all** $t \in \{1, 2, \dots\}$ **do**
 4: Observe $x_t, \mathbf{E}(x_t, \theta_t)$
 5: Calculate gradient:

$$\nabla_{\theta} \mathbf{E}(x_t, \theta_t) = \left(\frac{\partial}{\partial \theta_t[1]} \mathbf{E}(x_t, \theta_t), \dots, \frac{\partial}{\partial \theta_t[P]} \mathbf{E}(x_t, \theta_t) \right)^T$$

6: Update parameters:

$$\theta_{t+1} = \theta_t - \alpha_t \nabla_{\theta} \mathbf{E}(x_t, \theta_t)$$

算法 15 随机梯度下降

219

其中, $\mathbf{E}_i(x_i, \theta_i)$ 是第 i 个输入 x_i 的误差, $\alpha_t \in [0, 1]$ 是步长参数。如果误差在单一的输入-输出对上定义, 这种更新叫作随机梯度下降更新法。批量更新可以在离线算法中使用, 随机梯度下降更新更适合在线算法。

有迹象表明, 通常随机梯度下降比批量梯度下降收敛得更快 [Wilson and Martinez, 2003]。另外一个随机梯度下降之于批量梯度下降的优势是, 能够直接扩展在线随机梯度下降到非平稳目标, 例如, 如果策略在一次更新后改变。这些特征使得在线梯度方法十分适用于在线强化学习。一般而言, 虽然在一些情况下可以证明收敛到一个局部最优解 [Maei et al, 2009], 但是不能确保将强化学习的收敛性关联到一个最优解。

在神经网络中, 梯度下降往往通过反向传播来实现 [Bryson and Ho, 1969; Werbos, 1974; Rumelhart et al, 1986], 反向传播时, 使用链式法则和网络层结构有效地计算网络的输出关于它的参数的导数。然而, 梯度下降法的原则可以应用于任何可微函数。

在某些情况下, 正常的梯度不是最好的选择。更正式地说, 由于参数间的相互作用, 一般的梯度下降问题是参数空间中的距离度量可能与函数空间中的距离度量不同。令 $d\theta \in \mathbb{R}^P$ 表示参数空间的一个向量。该向量的欧几里得范数是 $\|d\theta\| = d\theta^T d\theta$ 。然而, 如果参数空间是一个弯曲的空间 (称作黎曼流形), 使用 $d\theta^T G d\theta$ 是更合适的, 其中, G 是一个 $P \times P$ 的半正定矩阵。使用这种加权距离度量, 最速下降方向变为

$$\tilde{\nabla}_{\theta} \mathbf{E}(x, \theta) = G^{-1} \nabla_{\theta} \mathbf{E}(x, \theta)$$

称作自然梯度 [Amari, 1998]。总之, 矩阵 G 最好的选择依赖于 \mathbf{E} 的函数形式。由于 \mathbf{E} 通常是未知的, G 通常需要估计。

自然梯度下降具有一些优势。例如, 自然梯度关于参数转换是不变的。换句话说, 当使用自然梯度时, 函数中的变化不依赖于函数的精确的参数化。这有点类似于在 7.2.1.2 节中的观察, 可以在价值空间而不是参数空间放缩步长来调整步长大小。只有在这里, 我们考虑参数更新的方向, 而不是它的大小。此外, 自然梯度避开了函数空间中的平稳时期, 通常会使得收敛更快。在 7.3.2.1 节中, 讨论策略-梯度算法时, 会更详细地讨论自然梯度。

7.2.3.2 无梯度优化

当被优化的函数是不可微的或者当预期会有很多局部最优存在时, 无梯度方法是有用的。有许多通用的全局最优化方法, 包括进化算法 [Holland, 1962; Rechenberg, 1971; Holland, 1975; Schwefel, 1977; Davis, 1991; Bäck and Schwefel, 1993]、模拟退火 [Kirkpatrick, 1984]、粒子群优化 [Kennedy and Eberhart, 1995] 和交叉熵优化 [Rubinstein,

220

1999; Rubinstein and Kroese, 2004]。这些方法中的大部分有一些共同的特点, 我们将在下面概述。我们关注于交叉熵和进化算法的一个子集, 但是, 可以十分类似地使用其他方法。对于进化算法的介绍可参见 [Bäck, 1996; Eiben and Smith, 2003]。关于进化算法在强化学习中的一个更广泛的描述参见第 10 章。我们简要介绍这些算法的工作原理。

这里描述的所有方法使用一个解决方案的种群。传统的进化算法建立一个解决方案种群并通过选择一些方案适应这个种群, 重组这些变异和可能变异的结果。新获得的方案替换旧的种群中的一些或所有方案。选择过程通常考虑到方案的适应度, 使得具有更高质量的方案有更大的概率用来建立新的方案。

最近, 适应生成解决方案的概率分布的参数而不是解决方案本身变得更常见。这种方法被用在进化策略中 [Bäck, 1996; Beyer and Schwefel, 2002]。这种方法生成一个种群, 使用方案的适应度来适应生成分布的参数, 而不是方案本身。然后, 通过从适应的概率分布生成新的方案, 获得了一个新的种群。一些具体的算法包括以下内容。协方差矩阵的适应进化策略 (CMA-ES) [Hansen and Ostermeier, 2001] 根据它们的适应度权衡采样的解决方案并使用加权平均作为新分布的均值。自然的进化策略 (NES) [Wierstra et al, 2008; Sun et al, 2009] 使用所有生成的方案来估计生成函数的参数的梯度, 然后使用自然梯度下降来优化这些参数。交叉熵优化方法 [Rubinstein and Kroese, 2004] 简单地选择 m 个具有最高适应度的解决方案 (其中 m 是一个参数), 并使用这些解决方案的均值寻找分布的新均值[⊖]。

221

```

1: input: parametrized population PDF  $p: \mathbb{R}^K \times \mathbb{R}^P \rightarrow \mathbb{R}$ , fitness function  $f: \mathbb{R}^P \rightarrow \mathbb{R}$ ,
   initial parameters  $\zeta_0 \in \mathbb{R}^K$ , population size  $n$ 
2: output: a parameter vector  $\zeta$  such that if  $\theta \sim p(\zeta, \cdot)$  then  $f(\theta)$  is large with high
   probability
3: for all  $t \in \{1, 2, \dots\}$  do
4:   Construct population  $\Theta_t = \{\bar{\theta}_1, \bar{\theta}_2, \dots, \bar{\theta}_n\}$ , where  $\bar{\theta}_i \sim p(\zeta_t, \cdot)$ 
5:   Use the fitness scores  $f(\bar{\theta}_i)$  to compute  $\zeta_{t+1}$  such that  $E\{f(\theta)|\zeta_{t+1}\} > E\{f(\theta)|\zeta_t\}$ 

```

算法 16 一个通用进化策略

一个通用进化策略如算法 16 所示。计算下一个参数的方法时, 在第 5 行动作生成函数设置 ζ_{t+1} 对于不同算法是不同的。然而, 尽量增加期望适应度, 使得 $E\{f(\theta)|\zeta_{t+1}\}$ 比之前的种群 $E\{f(\theta)|\zeta_t\}$ 期望适合性更高。这些期望定义为 $E\{f(\theta)|\zeta\} = \int_{\mathbb{R}^P} p(\zeta, \theta) f(\theta) d\theta$ 。

为了防止过早收敛到次优解, 应该注意分布的方差不要快速变小。一个简单的方式是, 为了防止每次迭代产生过大的变化, 在参数方面使用步长参数 [Rubinstein and Kroese, 2004]。为了防止过早收敛, 更复杂的方法包括: 使用 NES 的自然梯度、使用 CMA-ES 连续种群的协方差矩阵间的强相关性。

对于进化策略的最优解, 不能给出一般的收敛性保证。对于非稳定问题, 收敛到最优解 (例如强化学习中的控制问题) 看起来更难证明。尽管缺乏保证, 这些方法在实践中表现良好。主要的瓶颈通常是适应度的计算可能既嘈杂又昂贵。此外, 这些方法主要用来处理稳定

⊖ 根据这个描述, 可以认为交叉熵优化是一种类似于 CMA-ES 的进化策略, 使用了一种特殊的加权, 最好的 m 个解决方案权重分别为 $1/m$, 其余的为 0。然而, 已知的算法实现之间存在更多的差异, 其中, 最重要的也许对于新形成的呈 CMA-ES 分布的协方差矩阵的更优雅的估计, 目标是增加找到新的良好解决方案的概率。一些版本的交叉熵通过向方差里添加噪声来防止过早收敛 (例如, [Szita and Lőrincz, 2006]), 但是, 与 CMA-ES 使用的协方差估计相比, 这背后的理论似乎不太先进。最近研究已表明, CMA-ES 和 NES 是等效的, 除了在提出的算法实现中的一些差异 [Akimoto et al, 2011]。

的优化问题。因此，与逼近未知的最优策略的值相比，它们更适合于优化使用蒙特卡罗采样的策略。在7.4节中，我们比较CMA-ES和actor-critic时序差分方法的性能。

前面提到的无梯度方法都属于启发式的[Glover and Kochenberger, 2003]。这些方法迭代地搜索好的候选解集，或者产生这些解的一个分布。另一种方法是构建一个更易解（例如二次的）的需要优化的函数的模型，然后以分析的方法（例如，参见[Powell, 2002, 2006; Huyer and Neumaier, 2008]）最大化这一模型。新的样本可以反复选择以改进近似模型。我们知道还没有任何论文在强化学习中使用过这种方法，但是，在高维度问题中，这些方法的采样效率使得其自身成为一个将来研究中有趣的方向。

222

7.3 近似强化学习

在本节，我们将7.2节描述的通用的函数逼近技术应用到强化学习中。我们讨论连续域中最新的强化学习的当前状态。正如本章前面提到的，我们将不讨论逼近模型的构造，因为即使一个模型是已知的，在连续空间中具体计划往往是不可行的。

7.3.1 数值逼近

在数值逼近算法中，使用经验样本来更新给出当前或最优策略的近似的价值函数。很多强化学习算法属于这一类。这一类算法中，其主要区别是它们是在线还是离线更新，以及它们是开策略还是闭策略。最后，一个数值逼近算法可以存储一个状态-价值函数 $V: S \rightarrow \mathbb{R}$ 或者动作-价值函数 $Q: S \times A \rightarrow \mathbb{R}$ ，或者二者兼而有之[Wiering and van Hasselt, 2009]。我们将解释这些属性并给出对于每个属性组合的算法的举例。

开策略算法逼近状态-价值函数 V^π 或者动作-价值函数 Q^π ，它表示当前策略 π 的值。虽然最优策略 π^* 最初是未知的，但是，通过使用策略迭代，这些算法可以最终逼近最优价值函数 V^* 或者 Q^* ，在评估步骤之间改进策略。这种策略改进可能发生在每个时间步。闭策略算法可以学习不同策略的值，而不是其后接着的策略。这是有用的，因为这意味着我们不用必须跟随一个（接近）最优策略来学习最优策略的值。

在线策略在每个观察到的样本之后适应它们的数值逼近。离线策略对成批的样本进行操作。通常，在线算法每个样本需要的计算量更少，离线算法需要更少的样本来达到这一逼近的相似精度。

在线开策略算法包括时序差分（TD）算法，例如TD-学习[Sutton, 1984, 1988]、SARSA[Rummery and Niranjan, 1994; Sutton and Barto, 1998]和Expected-SARSA[van Seijen et al, 2009]。

离线开策略算法包括最小二乘法，例如最小二乘时序差分（LSTD）[Bradtke and Barto, 1996; Boyan, 2002; Geramifard et al, 2006]，最小二乘策略评估（LSPE）[Nedić and Bertsekas, 2003]和最小二乘策略迭代（LSPI）[Lagoudakis and Parr, 2003]。由于有限的篇幅，我们将不在本章讨论最小二乘法，但请参见第3章。

223

可以说最知名的无模型在线闭策略算法是Q学习[Watkins, 1989; Watkins and Dayan, 1992]。它的衍生包括Perseus[Spaan and Vlassis, 2005]、延迟的Q学习[Strehl et al, 2006]和贝叶斯Q学习[Dearden et al, 1998以及参见第11章]。所有的这些变体试图通过使用贝尔曼最优方程的变体去估计最优策略。在一般情况下，闭策略算法不需要估计最优策略，但是也可以逼近任意其他策略[Precup et al, 2000; Precup and Sutton, 2001; Sutton et al, 2008; van

Hasselt, 2011]。Q 学习的离线变体包括拟合 Q 迭代 [Ernst et al, 2005; Riedmiller, 2005; Antos et al, 2008a]。

Q 学习的在线和离线变体都有一个就是数值逼近中的噪声，由于问题的随机性和函数逼近器的局限性，可能导致一个结构化的高估偏差。简短地说，Q 学习中 $\max_a Q_i(s, a)$ 的值，可能在预期中远大于 $\max_a Q^*(s, a)$ 。这种偏差会严重减慢 Q 学习的收敛，即使在表格的设置中 [van Hasselt, 2010]，如果没有留意函数逼近器的选择，可能导致参数的发散 [Thrun and Schwartz, 1993]。一个针对这种偏移的不完全的解决方案由双 Q 学习 (van Hasselt, 2010) 算法给出，其中，两个动作 - 价值函数产生一个估计，这可能低估 $\max_a Q^*(s, a)$ ，但是在期望之内。

许多之前提到的算法既可用于在线也可用于离线算法，但是更适合其中之一。例如，拟合 Q 迭代通常用作离线算法，由于该算法的计算过于昂贵以至于不能在每个样本上运行。相反，在线算法可以存储观察到的样本并重用它们，就像再次以一种经验重放 [Lin, 1992] 的形式观察它们。最小二乘和拟合变体经常用作时序差分算法的离线版本。但是也有例外，例如，在线增量 LSTD 算法 [Geramifard et al, 2006, 2007]。

如果初始的策略不能容易地达到状态空间中有趣的地方，那么，在线算法的优势是，策略通常更新得更快，因为值的更新不会延迟，直到获得一个足够大的样本批次。这意味着在控制问题中，在线算法有时候更简单有效。

在接下来的两小节中，我们将详细讨论一些在预定义误差度量上使用梯度下降更新的离线数值逼近算法。

7.3.1.1 目标函数

为了使用梯度下降更新一个值，我们必须选择一些可以最小化的误差度量方法。这类度量通常称为目标函数。为了能更形式化地解释这些目标函数，我们引入了函数空间和投影的概念。回想 \mathcal{V} 是使得 $V \in \mathcal{V}$ 的价值函数空间。令 $\mathcal{F} \subseteq \mathcal{V}$ 表示对于某个函数逼近器来说，可描述的函数空间。直观上，如果 \mathcal{F} 包含 \mathcal{V} 的一个大的子集，那么该函数是灵活的，并且可以精确逼近许多价值函数。然而，可能容易过拟合这些感知数据，并且，由于通常一个更灵活函数需要更多可调的参数，可能使得更新很慢。相反，如果 \mathcal{F} 比 \mathcal{V} 小，该函数不会很灵活。例如，一个线性逼近器的函数空间通常比一个非线性逼近器的小。一个参数化的函数有一个参数向量 $\theta = \{\theta[1], \dots, \theta[D_\theta]\} \in \mathbb{R}^{D_\theta}$ ，该向量在训练中可以调整，从而，函数空间定义为

$$\mathcal{F} = \{V(\cdot, \theta) \mid \theta \in \mathbb{R}^{D_\theta}\}$$

从这里起，如果我们想要强调时间依赖性，我们用 V_t 表示参数化的值，如果我们想强调参数依赖性，用 V^θ 表示。根据定义， $V_t(s) = V(s, \theta_t)$ 且 $V^\theta(s) = V(s, \theta)$ 。

在某种范数下，投影 $\Pi: \mathcal{V} \rightarrow \mathcal{F}$ 是将一个价值函数映射到 \mathcal{F} 中的最近可描述函数的算子。该投影定义为

$$\|V - \Pi V\|_w = \min_{v \in \mathcal{F}} \|V - v\|_w = \min_{\theta} \|V - V^\theta\|_w$$

其中， $\|\cdot\|_w$ 是加权范数。假设该范数是二次的，满足。

$$\|V - V^\theta\|_w^2 = \int_{s \in \mathcal{S}} w(s) (V(s) - V^\theta(s))^2 ds$$

这意味着，该投影由逼近器函数的形式以及范数的权重确定。

令 $B = B^\pi$ 或者 $B = B^*$ ，依赖于我们是在逼近给定策略的值还是最优策略的值。通常，对于

整个状态空间，找到一个满足贝尔曼方程 $V^\theta = BV^\theta$ 的参数向量是不可能的，因为值 BV^θ 可能不能用所选的函数表示。相反，我们所能希望的最好的情况是一个满足公式 (7.7) 的参数向量

$$V^\theta = \Pi BV^\theta \quad (7.7)$$

它称作贝尔曼方程； Π 映射贝尔曼算子的结果到可以用函数逼近表示的空间。

在一些情况下，给出投影的一个封闭形式的表示是可能的 [Tsitsiklis and Van Roy, 1997; Bertsekas, 2007; Szepesári, 2010]。例如，考虑一个有 N 个状态的有限状态空间和一个线性函数 $V_i(s) = \theta_i^\top \phi(s)$ ，其中 $D_\phi = D_\phi \ll N$ 。令 $p_s = P(s_i = s)$ 表示期望的采样每个状态的稳定状态概率，将这些值存储在 $N \times N$ 对角矩阵 P 中。假设状态总是根据这些固定的概率采样。最后， $N \times D_\phi$ 维的矩阵 Φ 对于它的行中的所有状态保持该特征向量，使得 $V_i = \Phi \theta_i$ 且 $V_i(s) = \Phi_s \theta_i = \theta_i^\top \phi(s)$ 。然后，投影算子可以用 $N \times N$ 的矩阵

$$\Pi = \Phi(\Phi^\top P \Phi)^{-1} \Phi^\top P \quad (7.8)$$

当特征是线性独立的时，其逆存在， Φ 的秩为 D_ϕ 。

这一定义 $\Pi V_i = \Pi \Phi \theta_i = \Phi \theta_i = V_i$ ，但是 $\Pi B V_i \neq B V_i$ ，除非 $B V_i$ 可以表示为特征向量的线性函数。将像公式 (7.8) 的定义的投影矩阵用在分析和一些算法的推导中 [Tsitsiklis and Van Roy, 1997; Nedić and Bertsekas, 2003; Bertsekas et al, 2004; Sutton et al, 2008, 2009; Maei and Sutton, 2010]。我们在下一节讨论这些问题。

7.3.1.2 梯度时序差分学习

我们将标准的时序差分学习 (TD 学习) [Sutton, 1984, 1988] 泛化为一个函数逼近器的参数的梯度更新。表格的 TD 学习更新为

$$V_{t+1}(S_t) = V_t(S_t) + \alpha_t(S_t) \delta_t$$

其中， $\delta_t = r_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)$ ， $\alpha_t(S) \in [0, 1]$ 是一个步长参数。当 TD 学习在开策略更新的情况下，用来估计给定稳定策略的值时，并且特征向量是线性独立时，价值函数收敛 [Sutton, 1984, 1988]。之后表明，当使用资格迹并且特征不是线性独立时，TD 学习也收敛 [Dayan, 1992; Peng, 1993; Dayan and Sejnowski, 1994; Bertsekas and Tsitsiklis, 1996; Tsitsiklis and Van Roy, 1997]。最近有人提出了 TD 学习的变体，它在闭策略更新时是收敛的 [Sutton et al, 2008, 2009; Maei and Sutton, 2010]。下面我们讨论这些变体。大多数上述结果的限制是，它们只适用于预测设置。最近已经做了一些工作将这些分析扩展到控制设置。从而引出了贪婪 GQ 算法，将 Q 学习扩展到线性函数逼近，并在一些条件下克服了发散的风险 [Maei et al, 2010]。

当状态值存储在表中时，TD 学习可以理解为关于一步时序差分误差的随机梯度下降更新

$$\mathbf{E}(S_t) = \frac{1}{2} (r_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t))^2 = \frac{1}{2} (\delta_t)^2 \quad (7.9)$$

如果 V_t 是一个满足 $V_t(s) = V(s, \theta_t)$ 的参数化函数，关于参数的负梯度为

$$-\nabla_{\theta} (S_t, \theta) = -(r_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)) \nabla_{\theta} (r_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t))$$

除了状态和参数，误差也依赖于 MDP 和策略。我们不显式地详述这些依赖关系，从而避免符号的混乱。

基于公式 (7.9) 的误差梯度下降的直接实现是，调整参数来移动 $V_t(s)$ ，使它更接近 $r_{t+1} + \gamma V_t(S_{t+1})$ ，但是，也移动 $\gamma V_t(S_{t+1})$ ，使它更接近 $V_t(S_t) - r_{t+1}$ 。这样的算法叫作残差梯度算法

226

[Baird, 1995]。另外, 我们可以将 $r_{t+1} + \gamma V_t(s_{t+1})$ 理解为不依赖 q 的 V^* 的随机逼近。然后, 负梯度为 [Sutton, 1984, 1988]

$$-\nabla_{\theta} \mathbf{E}_t(s_t, \theta) = (r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)) \nabla_{\theta} V_t(s_t)$$

这表明参数可以更新为

$$\theta_{t+1} = \theta_t + \alpha_t(s_t) \delta_t \nabla_{\theta} V_t(s_t) \quad (7.10)$$

这是常规的 TD 学习更新, 它通常比残差梯度更新收敛得更快 [Gordon, 1995, 1999]。对于线性逼近, 对于任意 q , 有 $\nabla_{\theta} V_t(s_t) = \phi(s_t)$, 我们得到像公式 (7.4) 中的 tile 编码一样的更新。通过将公式 (7.10) 中的 $\nabla_{\theta} V_t(s_t)$ 替换为 $\nabla_{\theta} Q_t(s_t, a_t)$, 得到了对动作-值算法的相似的更新, 例如, 使用

$$\begin{aligned} \delta_t &= r_{t+1} + \gamma \max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t) \\ \text{或者 } \delta_t &= r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \end{aligned}$$

分别作为 Q 学习和 SARSA 的更新。

我们可以用下面的两个方程来合并累计资格迹和估计参数 λ [Sutton, 1984, 1988]:

$$\begin{aligned} e_{t+1} &= \lambda \gamma e_t + \nabla_{\theta} V_t(s_t) \\ \theta_{t+1} &= \theta_t + \alpha_t(s_t) \delta_t e_t + 1 \end{aligned}$$

其中, $e \in \mathbb{R}^{D_{\phi}}$ 是轨迹向量。替换轨迹 [Singh and Sutton, 1996] 是不太直接的, 虽然 [Främling, 2007] 的建议看起来是合理的:

$$e_{t+1} = \max(\lambda \gamma e_t, \nabla_{\theta} V_t(s_t))$$

由于这很好地对应于 tile 编码的常规做法, 当值存储在表里时, 这种更新归纳为传统的替换轨迹更新。然而, 对于该更新, 仍然缺乏一个很好的理论证明。

当使用闭策略时, 用公式 (7.10) 更新参数可能会发散。当使用线性 [Baird, 1995] 或者非线性函数逼近 [Tsitsiklis and Van Roy, 1996] 时, 对于 $\lambda < 1$ 的任何时序差分方法, 这个结论都成立。换句话说, 如果从一个不完全符合在估计策略情况下可能发生的状态-访问概率的分布中进行样本转换, 那么, 函数的参数可能发散。这是不幸的, 因为在控制设置中, 最终, 我们想要学习未知的最优策略。

最近, 有人提出了一类算法来处理该问题 [Sutton et al, 2008, 2009; Maei et al, 2009; Maei and Sutton, 2010]。这个想法是在二次投影的时序差分上进行随机梯度下降更新。

227

$$\mathbf{E}(\theta) = \frac{1}{2} \|V_t - \Pi B V_t\|_P = \frac{1}{2} \int_{s \in \mathcal{S}} P(s = s_t) (V_t(s) - \Pi B V_t(s))^2 ds \quad (7.11)$$

与公式 (7.9) 相比, 公式 (7.11) 的误差不依赖于时间步长或者状态。公式 (7.11) 中的范数依据存储在对角矩阵 P (如 7.3.1.1 节所述) 中的状态概率进行加权。如果最小化公式 (7.11), 我们达到公式 (7.7) 中的固定点。为此, 重写误差为

$$\mathbf{E}(\theta_t) = \frac{1}{2} (E\{\delta_t \nabla_{\theta} V_t(s)\})^T (E\{\nabla_{\theta} V_t(s) \nabla_{\theta}^T V_t(s)\})^{-1} E\{\delta_t \nabla_{\theta} V_t(s)\} \quad (7.12)$$

其中, 假设矩阵的逆存在 [Maei et al, 2009]。期望在 P 中的状态概率上取得。该误差是多个期望值的积。这些期望值不能只从一个实验中采集, 因为这样, 样本将是相关的。这可以通过更新一个附加的参数向量来解决。我们使用缩写 $\phi = \phi(s_t)$ 和 $\phi' = \phi(s_{t+1})$, 并且假设线性函数逼近。从而, $\nabla_{\theta} V_t(s_t) = \phi$, 得到

$$-\nabla_{\theta} \mathbf{E}(\theta_t) = E\{(\phi - \gamma \phi') \phi^T\} (E\{\phi \phi^T\})^{-1} E\{\delta \phi\}$$

$$\approx E\{(\phi - \gamma \phi') \phi^T\} \mathbf{w}$$

其中, $\mathbf{w}_t \in \mathbb{R}^{D_\phi}$ 是一个附加的参数向量。该向量应该逼近 $(E\{\phi \phi^T\})^{-1} E\{\delta_t \phi\}$, 可以用下面的更新来完成

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \beta_t(s_t)(\delta_t - \phi^T \mathbf{w}_t) \phi$$

其中, $\beta_t(s_t) \in [0, 1]$ 是一个步长参数。然后, 只剩下一个期望值需要逼近, 可以用单个样本完成。这导致了下面的更新

$$\theta_{t+1} = \theta_t + \alpha_t(s_t)(\phi - \gamma \phi')(\phi^T \mathbf{w}_t)$$

该更新称作 GTD2 (梯度差分学习, 版本 2) 算法 [Sutton et al, 2009]。也可以用一个稍微不同的方式写这个梯度来获得相似的 TDC 算法, 定义如下:

$$\theta_{t+1} = \theta_t + \alpha_t(s_t)(\delta_t \phi - \gamma \phi'(\phi^T \mathbf{w}_t))$$

其中 \mathbf{w}_t 如上更新。该算法叫作梯度相关的 (TDC) 的 TD, 因为对于主要参数向量 θ_t 的更新等于公式 (7.10), 除了一个相关项。在使用闭策略更新时, 这一项防止了参数的发散。可以证明, 如果状态根据 P 采样, 那么, GTD2 和 TDC 都可以最小化公式 (7.12)。与一般的 TD 学习不同的是, 当 P 中的概率不同于跟随策略 π 产生的结果时, 这些算法也是收敛的。这是有用的, 例如, 当有一个模拟器, 允许我们以任何顺序对状态进行采样的时候, 而 π 将会浪费更多的时间在不感兴趣的状态上。

当使用非线性光滑函数逼近器时, 可以证明, 相似的算法达到局部最优 [Maei et al, 2009]。对非线性算法的更新与上面提到的方法相似, 将使用另一个相关项。为了用资格迹学习动作值, 这些更新可以扩展为 Q 学习的形式。作为结果的 GQ(λ) 算法是闭策略, 并且收敛到一个给定的估计策略的值, 甚至当算法服从一个不同的动作策略 [Maei and Sutton, 2010] 时。这些方法可以扩展到具有贪婪非稳定的估计策略的控制问题, 虽然目前尚不清楚产生的贪婪 GQ 算法在实际中表现如何。

[228]

虽然这些理论见解和所得到的算法是很有前景的, 但是, 实践中, 在开策略设置中, TD 更新公式 (7.10) 仍然是更好的选择。此外, 类似于公式 (7.10) 对 Q 学习的更新通常产生好的策略, 虽然通常不能保证收敛。此外, 对于特定的函数 (所谓的均衡器) Q 学习不收敛 [Gordon, 1995; Szepesvári and Smart, 2004]。实际上, 许多问题没有这些导致参数发散的明确的特点。最后, 收敛保证主要限于从固定的稳定状态概率中采样的使用。

如果可以最小化叫作贝尔曼的残差 $E(\theta_t) = \|V - B^*V\|_\rho$, 它会自动最小化公式 (7.11) 中预期的时序差分误差。将 $(\delta_t)^2$ 用作该误差 (其中 $B = B^*$) 的一个采样导致了有偏估计, 但是已经提出了使用该误差的其他方法 [Antos et al, 2008b; Maillard et al, 2010]。最小化残差是否比最小化预期误差产生更好的结果是取决于问题的 [Scherrer, 2010]。

扩展标准的在线时序差分算法 (例如 Q 学习) 到连续的动作空间是不平凡的。虽然可以对每一个连续动作构建一个值的评估, 但是, 当有无限多的动作时, 快速找到最大化的动作是平凡的。一种解决方案是简单地离散化动作空间, 就像在 tile 编码中, 或者通过执行线搜索 [Pazis and Lagoudakis, 2009]。另一种方法是使用内插器, 例如, 线拟合 [Baird and Klopff, 1993; Gaskett et al, 1999] 中, 对于每个状态, 输出候选动作-值对的一个固定的数目。对这些动作和值进行插值, 在当前状态下, 形成一个对连续的动作-价值函数的估计。由于插值, 所得函数的最大值将总是精确地处于候选动作之一, 从而在连续空间中, 促进贪婪动作的选择。然而, 下一节中的算法通常更好地适用于连续动作问题。

7.3.2 策略逼近

正如讨论的那样，从分析上来看，由一个模型确定一个好的策略可能是困难的。一个近似的状态-动作价值函数 Q 使这变得更简单，因为在每个状态 s 中，贪婪策略可以通过选择最大化 $Q(s,a)$ 的参数 a 来找到。然而，如果动作空间是连续的，那么，在每个状态中找到贪婪动作可能是非平凡的并且耗时的。因此，存储最优策略的一个明确的估计是有利的。在本节中，我们考虑存储参数化策略 $\pi : S \times A \times \mathcal{P} \rightarrow [0,1]$ 的 actor-only 和 actor-critic 算法，其中， $\pi(s,a,\psi)$ 表示对于给定的策略参数向量 $\psi \in \mathcal{P} \subseteq \mathbb{R}^{D_\psi}$ ，在 s 中选择 a 的概率。该策略称作为一个 actor。

在 7.3.2.1 节中，讨论策略-梯度算法的一般框架，以及怎样使用该框架来改进策略。在 7.3.2.2 节中，讨论进化策略对于直接策略搜索的应用。然后，在 7.3.2.3 节中，讨论使用该框架和价值函数逼近的 actor-critic 方法。最后，在 7.3.2.4 节中，讨论对 actor 使用不同更新方式的另一个 actor-critic 方法。

7.3.2.1 策略-梯度算法

策略-梯度算法的思想是，使用在累积期望值 V^π 上的梯度上升来更新策略 [Williams, 1992; Sutton et al, 2000; Baxter and Bartlett, 2001; Peters and Schaal, 2008b; Rückstieβ et al, 2010]。如果梯度是已知的，可以用以下方式更新策略

$$\psi_{k+1} = \psi_k + \beta_k \nabla_\psi E\{V^\pi(s_t)\} = \psi_k + \beta_k \nabla_\psi \int_{s \in S} P(s_t=s) V^\pi(s) ds$$

这里 $P(s_t=s)$ 表示学习器在状态 s 、时间步长 t 下的概率， $\beta_k \in [0,1]$ 是步长。在更新中，除了 t ，使用下 k 来区分动作的时间步长与策略参数的更新进度，它们是不可以重叠的。如果状态空间是有限的，那么我们用来替换积分。

作为一个可行的选择，我们可以使用随机梯度下降：

$$\psi_{t+1} = \psi_t + \beta_t(s_t) \nabla_\psi V^\pi(s_t) \quad (7.13)$$

这里，更新的时间步长对应于动作的时间步长，使用下标 t 。这种程序充其量能够找到一个局部最优，因为在通常情况下，它们使用关于策略参数非凸的价值函数的梯度。然而，得到了一些有希望的结果，例如，在机器人学中 [Benbrahim and Franklin, 1997; Peters et al, 2003]。

更新公式 (7.13) 的明显的问题是通常情况下 V^π 是未知的，因此，两者都不是它的梯度。对于一个成功的策略-梯度算法，我们需要 $\nabla_\psi V^\pi$ 的一个估计。我们将不讨论如何获得这样的一个估计。

我们将使用轨线的概念。轨线 \mathcal{S} 是一个状态和动作的序列：

$$\mathcal{S} = \{s_0, a_0, s_1, a_1, \dots\}$$

[229] 给定的轨线出现的概率等于其相应的状态和动作序列出现的概率：

$$\begin{aligned} P(\mathcal{S}|s, \psi) &= P(s_0 = s) P(a_0 | s_0, \psi) P(s_1 | s_0, a_0) P(a_1 | s_1, \psi) P(s_2 | s_1, a_1) \dots \\ &= P(s_0 = s) \prod_{t=0}^{\infty} \pi(s_t, a_t, \psi) P_{s_t, a_t}^{s_{t+1}} \end{aligned} \quad (7.14)$$

对于给定的策略，期望值 V^π 可以表示为所有可能的轨线上的积分，对应的期望回报为：

$$V^\pi(s) = \int_{\mathcal{S}} P(\mathcal{S} | s, \psi) E \left\{ \sum_{t=0}^{\infty} \gamma^t r_{t+1} | \mathcal{S} \right\} d\mathcal{S}$$

从而，梯度方法用封闭形式表示为：

$$\begin{aligned}
\nabla_{\psi} V^{\pi}(s) &= \int_{\mathcal{S}} \nabla_{\psi} P(\mathcal{S} | s, \psi) E \left\{ \sum_{t=0}^{\infty} \gamma^t r_{t+1} | \mathcal{S} \right\} d\mathcal{S} \\
&= \int_{\mathcal{S}} P(\mathcal{S} | s, \psi) \nabla_{\psi} \log P(\mathcal{S} | s, \psi) E \left\{ \sum_{t=0}^{\infty} \gamma^t r_{t+1} | \mathcal{S} \right\} d\mathcal{S} \\
&= E \left\{ \nabla_{\psi} \log P(\mathcal{S} | s, \psi) E \left\{ \sum_{t=0}^{\infty} \gamma^t r_{t+1} | \mathcal{S} \right\} | s, \psi \right\}
\end{aligned} \tag{7.15}$$

其中，我们使用了普通恒等式 $\nabla_x f(x) = f(x) \nabla_x \log f(x)$ 。这一有用的观察与 Fisher 的打分函数 [Fisher, 1925; Rao and Poti, 1946] 和似然比 [Fisher, 1922; Neyman and Pearson, 1928] 相关。在提出策略-梯度算法（参见 [Peters and Schaal, 2008b]）之后，[Williams, 1992] 将其应用到强化学习中，为此，有时将其称为 REINFORCE 技巧。

在如公式 (7.14) 中所给的轨线概率的定义的结果表明公式 (7.15) 中的对数包括一个部分的和，在其中，只有策略部分依赖于 ψ 。因此，当采取梯度方法时，其他的部分消失了，得到：

$$\begin{aligned}
\nabla_{\psi} \log P(\mathcal{S} | s, \psi) &= \nabla_{\psi} \left(\log P(s_0 = s) + \sum_{t=0}^{\infty} \log \pi(s_t, a_t, \psi) + \sum_{t=0}^{\infty} \log P_{s_t, a_t}^{r_{t+1}} \right) \\
&= \sum_{t=0}^{\infty} \nabla_{\psi} \log \pi(s_t, a_t, \psi)
\end{aligned} \tag{7.16}$$

这很不错，因为这表明我们不需要转换模型。然而，这只在策略是随机的时候成立。如果策略是确定的，那么我们需要梯度 $\nabla_{\psi} \log P_{sa}^{s'} = \nabla_a \log P_{sa}^{s'} \nabla_{\psi} \pi(s, a, \psi)$ ，它只在当转换概率已知时可用。在大多数情况下，这不是一个大的问题，因为无论如何都需要随机策略来确保充分的搜索。图 7.3 展示了两个可以使用的随机策略的示例和对应的梯度。

231

玻尔兹曼搜索可以用在离散动作空间中。假设 $\phi(s, a)$ 是一个大小为 D_{ψ} 的特征向量，对应状态 s 和动作 s 。假设策略是一个参数为 ψ 的玻尔兹曼分布，使得

$$\pi(s, a, \psi) = \frac{e^{\psi^T \phi(s, a)}}{\sum_{b \in A(s)} e^{\psi^T \phi(s, b)}}$$

那么，该策略的对数梯度为

$$\nabla_{\psi} \log \pi(s, a, \psi) = \phi(s, a) - \sum_b \pi(s, b, \psi) \phi(s, b)$$

高斯搜索可以用在连续动作空间中。例如，一个均值为 $\mu \in \mathbb{R}^{D_A}$ ， $D_A \times D_A$ 大小的协方差矩阵 Σ ，使得

$$\pi(s, a, \{\mu, \Sigma\}) = \frac{1}{\sqrt{2\pi \det \Sigma}} \exp \left(-\frac{1}{2} (a - \mu)^T \Sigma^{-1} (a - \mu) \right),$$

$$\nabla_{\mu} \log \pi(s, a, \{\mu, \Sigma\}) = (a - \mu)^T \Sigma^{-1},$$

$$\nabla_{\Sigma} \log \pi(s, a, \{\mu, \Sigma\}) = \frac{1}{2} \left(\Sigma^{-1} (a - \mu) (a - \mu)^T \Sigma^{-1} - \Sigma^{-1} \right).$$

其中，动作 $a \in A$ 是与 μ 大小相同的向量。如果 $\psi \in \Psi \subseteq \mathbb{R}^{D_{\psi}}$ 是决定均值 $\mu(s, \psi)$ 的状态依赖位置的参数向量，那么 $\nabla_{\psi} \log \pi(s, a, \psi) = J_{\psi}^T(\mu(s, \psi)) \nabla_{\mu} \log \pi(s, a, \{\mu, \Sigma\})$ ，其中 $J_{\psi}(\mu(s, \psi))$ 是 $D_A \times D_{\psi}$ 的雅克比矩阵，包含对于 $\mu(s, \psi)$ 的每个元素和 ψ 的每个元素的偏导。

协方差矩阵同样可以是参数化函数的输出，但是，需要注意保持充分的搜索。由于常规梯度可能很快地降低搜索，一种方式是使用自然梯度更新，另一个选择是使用协方差矩阵 $\sigma^2 I$ ，其中 σ 是一个可调的参数，它可以是固定的或者是根据预定义的模式递减的。

图 7.3 策略-梯度算法的随机策略示例

当知道公式 (7.16) 中的梯度, 我们可以在公式 (7.15) 中定量采样。对此, 我们需要取期望累积折扣回报的样本。例如, 如果任务是阶段性的, 我们可以取对于每个阶段给出累积 (可能是折扣的) 回报的蒙特卡罗样本。在阶段性的 MDP 中, 公式 (7.16) 中的和是有限的, 而不是无限的, 得到

$$\nabla_{\psi} V^{\pi}(s_t) = E \left\{ R_k(s_t) \left(\sum_{j=t}^{T_k-1} \nabla_{\psi} \log \pi(s_j, a_j, \psi) \right) \right\} \quad (7.17)$$

其中, $R_k(s_t) = \sum_{j=t}^{T_k-1} \gamma^{t-j} \gamma_{j+1}$ 是在阶段 k 达到状态 s_t 后, 得到的全部 (折扣的) 的返回, 其中, k 以 T_k 结束。可以通过公式 (7.13) 对梯度采样并用于策略更新。

从公式 (7.17) 采样的缺点是 $R_k(s_t)$ 的方差可能非常大, 导致梯度的嘈杂的估计。[Williams, 1992] 指出, 通过使用以下更新, 可以得到一定程度的缓和:

$$\psi_{t+1} = \psi_t + \beta_t(s_t)(R_k(s_t) - b(s_t)) \sum_{j=t}^{T_k-1} \nabla_{\psi} \log \pi(s_j, a_j, \psi_t) \quad (7.18)$$

其中, $b(s_t)$ 是不依赖策略参数的基线, 虽然, 它可能依赖于状态。该基线可以用来最小化方差, 而不会对更新添加偏差, 因此对所有的 $s \in S$

$$\begin{aligned} \int_{\mathcal{S}} \nabla_{\psi} P(\mathcal{S} | s, \psi) b(s) d\mathcal{S} &= b(s) \nabla_{\psi} \int_{\mathcal{S}} P(\mathcal{S} | s, \psi) d\mathcal{S} \\ &= b(s) \nabla_{\psi} 1 = 0 \end{aligned}$$

已经表明, 设置该基线等于状态值的一个估计, 使得 $b(s) = V_t(s)$, 可能是一个好主意 [Sutton et al, 2000; Bhatnagar et al, 2009], 虽然严格来说, 它关于策略参数不再是独立的。已经做了一些工作以最佳的方式设置基线来最小化方差, 从而, 提高了算法的收敛速度 [Greensmith et al, 2004; Peters and Schaal, 2008b], 但是, 我们在此不详述这个问题。

上述定义的策略-梯度更新使用一个沿着性能度量的最陡上升方向更新策略参数的梯度。然而, 梯度更新在参数空间进行操作, 而不是策略空间。换句话说, 当使用常规梯度下降和一个步长时, 我们在参数空间 $d\psi_t^T d\psi_t$ 中限制变化的大小, 其中, $d\psi_t = \psi_{t+1} - \psi_t$ 是参数中的变换。有人认为, 在策略空间中限制步长是更好的。这类似于在 7.2.1.2 节中的观察, 在参数空间中, 对于一个线性函数逼近器的更新可能导致在具有预料之外的大或小的步长的值空间的更新。对于策略的一个好的距离度量方式是 Kullback-Leibler 散度 [Kullback and Leibler, 1951; Kullback, 1959]。这可以用一个二阶泰勒展开 $d\psi_t^T F_{\psi} d\psi_t$ 来逼近, 其中 F_{ψ} 是 $D_{\psi} \times D_{\psi}$ 的 Fisher 信息矩阵, 定义为

$$F_{\psi} = E \left\{ \nabla_{\psi} P(\mathcal{S} | s, \psi) \nabla_{\psi}^T P(\mathcal{S} | s, \psi) \right\}$$

它的期望涵盖了可能的轨线。可以使用公式 (7.16) 来从该矩阵中采样。然后, 可以获得一个自然的策略梯度, 它服从一个自然的梯度 [Amari, 1998]。[Kakade, 2001] 第一次将这一思想引入到强化学习中。要求的更新因此变成了

$$\psi_{t+1}^T = \psi_t^T + \beta_t(s_t) F_{\psi}^{-1} \nabla_{\psi} V^{\pi}(s_t) \quad (7.19)$$

再对其进行采样。这种更新的一个缺点是需要足够的样本来 (近似地) 计算矩阵的逆 F_{ψ}^{-1} 。如果参数数量相当大, 那么, 所需样本的数量可能是受限的, 除非采样由一个可以使用很多时间步长来完成的阶段组成。

使用自然梯度的大多数算法在每次更新时需要花费 $O(D_{\psi}^2)$ 时间, 并且可能需要合理的样

本量。更多细节详见 [Kakade, 2001; Peters and Schaal, 2008a; Wierstra et al, 2008; Bhatnagar et al, 2009; Rückstieß et al, 2010]。

7.3.2.2 策略搜索与进化策略

如果不是策略参数上的梯度更新，我们还可以在策略-参数空间构建一个无梯度搜索。举例来说，结合自然策略-梯度和进化策略的思想，我们讨论自然进化策略 (NES) [Wierstra et al, 2008; Sun et al, 2009]。该算法背后的思想相当简单，虽然许多具体的改进更加先进 [Sun et al, 2009]。在 7.2.3.2 节中讨论的其他的无梯度方法可以用在相似的情况下。

NES 创建一个 n 参数的向量 ψ_1, \dots, ψ_n 种群，而不是存储单个搜索策略。这些向量表示具有某种期望收益的策略。这一收益可以通过蒙特卡罗采样 $R_k(s_0)$ 来取样，这与公式 (7.17) 相类似，其中 s_0 是一个阶段中的第一个状态。蒙特卡罗采样是适当的。目标是改进产生这些策略参数分布的种群参数，使得新的种群分布有可能生成更好的策略。与策略-梯度方法相比，我们没有改进策略本身，我们改进产生这些策略的过程。为此，我们在当前解的适应度上，使用梯度上升步骤。

在 NES 和 CMA-ES 中，参数向量 ψ_i 来自一个高斯分布 $\psi_i \sim \mathcal{M}(\mu_\psi; \Sigma_\psi)$ 。对于均值和协方差矩阵，令 ζ_ψ 为包含所有种群参数的向量。对于 μ_ψ 和 Σ_ψ 中的种群参数，NES 使用蒙特卡罗采样来找到性能的自然梯度 $F_\zeta^{-1} \nabla_\zeta E\{R\}$ 的估计。这告诉我们，为了在将来生成更好的种群，原参数应该如何改变。由于高斯生成分布的选择，以分析的方式计算 Fisher 信息矩阵是可能的。加上进一步的算法细节，在 NES 中，将单代的计算量限制在 $O(np^3 + nf)$ 内是可能的，其中 n 是种群中解决方案的数量， p 是一个解决方案的参数的数量， f 是对于单一解确定适应度的计算代价。注意，如果必须的蒙特卡罗 roll-outs 是长的， f 可能很大。适应度中潜在的大的方差可能使得直接策略搜索不太适用于大的有噪声的问题。注意，与策略-梯度算法相比，候选策略可能是确定的，这在某种程度上将削弱方差。

7.3.2.3 actor-critic 算法

公式 (7.17) 中，如果使用蒙特卡罗 roll-outs，估计 $\nabla_\psi V^\pi(s_t)$ 的方差可能很大，这将严重减慢收敛。同样，这对于使用蒙特卡罗 roll-outs 的直接策略-搜索算法也是可能的。这个问题 [234] 的一个潜在的解决方案可以使用 V^π 的显式逼近来表示。在此背景下，这样的近似价值函数称作一个 critic，这个组合的算法称为 actor-critic 算法 [Barto et al, 1983; Sutton, 1984; Konda and Borkar, 1999; Konda, 2002; Konda and Tsitsiklis, 2003]。

actor-critic 算法通常使用时序差分算法来更新 V_t ，它是 V^π 的一个估计。可以表明，如果 a_t 根据 π 来选择，在一些假设下，TD 误差 $\delta_t = r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)$ 是 $Q^\pi(s_t, a_t) - V^\pi(s_t)$ 的无偏估计。假设以 $b(s_t) = V^\pi(s_t)$ 为基线，对于策略梯度，这导致了一个 $\delta_t \nabla_\psi \log \pi(s_t, a_t, \psi_t)$ 的无偏估计 [Sutton et al, 2000]。该估计可扩展到一个近似的自然梯度方向 [Peters et al, 2003; Peters and Schaal, 2008a]，得出自然 actor-critic (NAC) 算法。一个典型的 actor-critic 更新将使用以下方式更新策略参数，

$$\psi_{t+1} = \psi_t + \beta_t(s_t) \delta_t \nabla_\psi \log \pi(s_t, a_t, \psi_t)$$

其中对于一个 NAC 算法， $\nabla_\psi \log \pi(s_t, a_t, \psi_t)$ 可以用 $F_\psi^{-1} \nabla_\psi \log \pi(s_t, a_t, \psi_t)$ 代替。

在一些情况下，不需要显式地逼近 Fisher 逆信息矩阵，而通过使用线性函数逼近器 $g_t^\pi(s, a, w) = w_t^\top \nabla_\psi \log \pi(s, a, \psi_t)$ 来逼近 $Q^\pi(s, a) - b(s)$ [Sutton et al, 2000; Konda and Tsitsiklis, 2003; Peters and Schaal, 2008a]。经过一些代数操作之后，得到

$$\nabla_{\psi} V_t(s) = E \left\{ \nabla_{\psi} \log \pi(s, a, \psi_t) \nabla_{\psi}^T \log \pi(s, a, \psi_t) \right\} w_t = F_{\psi} w_t$$

将它插入公式 (7.19) 得到 NAC 更新

$$\psi_{t+1} = \psi_t + \beta_t(s_t) w_t$$

然而, 这个简洁的更新只适用于使用特殊的线性函数形式 $g_t^{\pi}(s, a, w)$ 的 critic 来逼近 $Q^{\pi}(s, a) - b(s)$ 的值。而且, 更新的准确性明确地依赖于 w_t 的准确性。[Bhatnagar et al, 2009] 描述了其他的 NAC 变体。

本节中的一些策略 - 梯度想法与相关领域中的自适应动态规划 (ADP) [Powell, 2007; Wang et al, 2009] 的许多想法有明显的重叠。基本上, 可以认为强化学习和 ADP 是相同研究领域中的不同的名字。然而, 实际上, 在这类问题和提出的解决方案之间存在一个分歧。通常, 在自适应动态规划中采用了更多的工程角度, 这产生了一个有些不同的符号和一个有些不同的目标集。例如, 在 ADP 中, 目标通常是稳定布置 [(Murray et al, 2002)]。这给搜索带来了一些可以安全使用的限制, 并且表示通常目标状态是开始状态, 目标保持接近这个状态, 而不是找到更好的状态。此外, 连续时间中的问题比强化学习 [Beard et al, 1998; Vrabie et al, 2009] 中的问题讨论得更多, 强化学习使用了贝尔曼最优方程的连续版本, 称作汉密尔顿-雅克比-贝尔曼方程 [Bardi and Dolcetta, 1997]。对这些细节的进一步讨论不在本章的范围内。

最早的 actor-critic 方法之一来自 ADP 的文献。它逼近 Q^{π} 而不是 V^{π} 。假设使用高斯搜索, 以一个确定的函数 $Ac: S \times \mathcal{P} \rightarrow A$ 的输出为中心。这里, 我们把该函数叫作 actor, 而不是把整个策略作为 actor。如果我们使用一个可微函数 Q_t 来逼近 Q^{π} , 使用下面的链式法则来更新该 actor 的参数成为了可能,

$$\begin{aligned} \psi_{t+1} &= \psi_t + \alpha_t \nabla_{\psi} Q_t(s_t, Ac(s, \psi), \theta) \\ &= \psi_t + \alpha_t J_{\psi}^T(Ac(s_t, \psi)) \nabla_a Q_t(s_t, a) \end{aligned}$$

其中, $J_{\psi}(Ac(s, \psi))$ 是 $D_A \times D_{\mathcal{P}}$ 的 Jacobian 矩阵, 其第 i 行第 j 列的元素等于 $\frac{\partial}{\partial \psi_j} Ac_i(s, \psi)$, 其中, $Ac_i(s, \psi)$ 是 $Ac(s, \psi)$ 的第 i 个元素。该算法称为动作依赖的启发式动态规划 (ADHDP) [Werbos, 1977; Prokhorov and Wunsch, 2002]。实际上, 可以用任何动作-值算法更新 critic, 包括 Q 学习, 表明是 Q^* 的一个估计而不是 Q^{π} 。这个算法有很多变体, 其中很多假设环境的一个已知模型、回报函数或者两者, 或者它们构建这样的模型。然后, 通常一个附加的假设是模型是可微的。

7.3.2.4 连续的 actor-critic 学习自动机

本节中, 我们讨论连续的 actor-critic 学习自动机 (Cacla) 算法 [van Hasselt and Wiering, 2007, 2009]。与大多数其他 actor-critic 方法相比, Cacla 使用动作空间中的误差而不是参数或策略空间, 它使用时序差分误差的符号, 而不是它的大小。

在 Cacla 算法中, 一个 critic $V: S \times \Theta \rightarrow \mathbb{R}$ 逼近 V^{π} , 其中 π 是当前策略。一个 actor $Ac: S \times \mathcal{P} \rightarrow A$ 对每个状态输出单一 (可能多维) 动作。在学习过程中, 假设存在搜索, 满足 $a_t \neq Ac(s_t, \psi_t)$, 其原因将很快变得清晰。例如, $\pi(s_t, \psi_t)$ 可以是一个以 $Ac(s_t, \psi_t)$ 为中心的高斯分布。就像在许多其他的 actor-critic 算法中一样, 如果时序差分误差 δ_t 是正的, 那么, 我们判断 a_t 是一个好的选择, 并且强化它。在 Cacla 中, 这将通过更新接近 a_t 的 actor 的输出来完成。这就是为什么搜索是必要的: 如果没有搜索, actor 输出已经等于动作, 参数不会被

更新[⊖]。

对 actor 的更新只发生在时序差分误差是正的情况下。这与对学习自动机 [(Narendra and Thathachar, 1974, 1989)] 的线性回报 – 不活跃更新是相似的, 这里的学习自动机使用时序差分误差的符号作为“成功”的度量。大多数其他的 actor-critic 方法使用时序差分误差的大小, 并且也在它的符号为负时, 沿着相反的方向更新。然而, 这对于 Cacla 通常不是一个好的思路, 因为这相当于向着某个尚未执行的动作进行更新, 对此, 是否比 actor 当前的输出更好是未知。作为一个极限的情况, 对于某个确定的 MDP, 在每个状态下, actor 已经输出了最优的动作。对于大多数搜索到的动作, 时序差分误差是正的。如果将 actor 更新到远离这样的动作, 它的输出将几乎不会是最优的。

[236]

这是 Cacla 和策略 – 梯度方法的一个重要的不同之处: 当观察到实际的改进时, Cacla 只更新它的 actor。当在值空间中存在稳定状态, 而且时序差分误差很小时, 这避免了慢学习。实证表明, 相比步长大小依赖于时序差分误差的情况, 这实际上产生了更好的策略 [van Hasselt and Wiering, 2007]。直观上, 与一个有希望的动作 a_t 的距离比数值上提高的大小更重要, 这是合情合理的。

Cacla 的一个基础版本见算法 17。第 3 行的策略可以依赖于 actor 的输出, 但是, 这不是严格必须的。例如, 动作空间中尚未搜索的部分可能受动作选择的青睐。在 7.4 节中, 我们将看到 Cacla 甚至可以从一个完全随机的策略中学习。Cacla 只在当 $a_t \neq Ac(s_t, \psi_t)$ 时, 才能更新它的 actor, 但在训练之后, 学习器可以确切地使用 actor 输出的动作。

第 6 行的 critic 更新是一个普通的 TD 学习更新。可以用以下的更新替换它: $TD(\lambda)$ 更新、增量的最小二乘更新或者 7.3.1.2 节中的其他更新。第 8 行的 actor 更新可以理解为执行的动作和 actor 输出之间的误差 $\|a_t - A_c(s_t, \psi_t)\|$ 的梯度下降。下面是第二个重要的与大多数其他 actor-critic 算法的不同之处: 直接使用动作空间中的误差更新策略, 而不是在参数空间 [Konda, 2002] 或者策略空间 [Peters and Schaal, 2008a; Bhatnagar et al, 2009] 更新策略。

[237]

在某些方面, Cacla 与进化策略相似。在强化学习的背景下, 进化策略通常存储参数空间的一个分布, 策略参数将从中采样。例如, NES [Rückstieß et al, 2010]、CMA-ES [Heidrich-Meisner and Igel, 2008] 和交叉熵优化 [Busoniu et al, 2010] 提出这种方法。相反, Cacla 使用动作空间中的概率分布: 动作 – 选择策略。

与策略 – 梯度算法相比, Cacla 与更多的搜索类型兼容。例如, 一个均匀随机策略将仍然允许 Cacla 更新它的 actor, 然而, 对于策略 – 梯度方法, 这样的策略没有参数调整。

在以往的工作中, Cacla 毫不逊色于 ADHDP 和线拟合 [van Hasselt and Wiering, 2007], 也可与离散时序差分方法 (例如, Q 学习和 Sarsa [van Hasselt and Wiering, 2009]) 媲美。在下一节中, 我们将它与 CMA-ES 和 NAC 在一个双极车杆的问题上进行比较。为简单起见, 在实验中, 我们使用算法 17 中概述的 Cacla 的简单版本。然而, Cacla 可以用很多方式进行扩展和改进。为了完整性, 在这里, 我们列出了一些可能的改进。

第一, Cacla 可以用资格迹进行扩展。例如, 价值函数可以用 $TD(\lambda)$ 或者新的变体 $TDC(\lambda)$ 和 $GTD2(\lambda)$ [Sutton et, 2009] 更新。后者的变体可能有利于用闭策略的方式为 actor 学习价值函数, 而不是为随机策略学习这个值, 该随机策略用来像这个算法的简单版本那样

⊖ 对与 actor 的输出相等的操作的反馈仍然可以改进价值函数。这可能是有用的, 因为当 actor 保持固定时, 价值函数可以改进。类似于策略迭代, 我们可以在没有搜索的情况下交错步骤来更新 critic, 并按步骤来更新 actor。虽然很有前景, 但是, 我们不在这里进一步探讨这种可能性。

搜索。这个 actor 也可以使用更新 actor 的输出的轨迹进行扩展。这样的决策轨迹是否提高了性能目前尚不可知。

第二, Cacla 可以使用批量更新进行扩展, 批量更新更有效地利用了在过去所观察到的经验。例如, 可以用(增量的)最小二乘时序差分学习 [Bradtke and Barto, 1996; Boyan, 2002; Geramifard et al, 2006] 或者(神经)拟合 Q 迭代 [Ernst et al, 2005; Riedmiller, 2005] 的一个变体。因为这样可以降低 TD 误差的方差, 所以可以防止把 actor 更新到一个更坏的动作, 并且可以允许更大的 actor 步长大小。相似地, 为了以重放 [Lin, 1992] 的方式重用样本, 将样本存储起来。

第三, Cacla 可以使用多 actor。这可以防止 actor 卡在一个局部最优策略上。然后, 可以使用离散算法(如 Q 学习)来在 actor 间直接进行选择。这种方法初步的结果是有希望的, 虽然附加的 actor 选择器引入了额外的需要调整的参数。

7.4 双极车杆实验

在本节中, 我们在一个双极车杆问题上, 比较 Cacla、CMA-ES 和 NAC。在这个问题中, 两个分开的杆被一个铰链系在一个手推车上。两个杆在长度和质量上不同, 并且两个杆必须通过打击车来保持平衡。

在强化学习中, 用了很多不同的度量来比较算法的性能, 不存在完全标准化的基准。因此, 我们根据一个早先的使用动力学的论文 [Heidrich-Meisner and Igel, 2008] 使用动力学和性能度量来比较 Cacla 与 CMA-ES 和 NAC 的结果。我们选择这篇论文的原因是, 它报告了应用 NAC 和 CMA-ES 的结果, 而两者是目前在直接策略搜索和黑盒优化 [Jiang et al, 2008; Gomez et al, 2008; Glasmachers et al, 2010; Hansen et al, 2010] 中最先进的方法。

双车杆的动力学如下 [Wieland, 1991]:

$$\ddot{x} = \frac{F - \mu_c \text{sign}(\dot{x}) + \sum_{i=1}^2 2m_i \dot{x}_i^2 \sin \chi_i + \frac{3}{4} m_i \cos \chi_i \left(2 \frac{u_i \dot{\chi}_i}{m_i l_i} + g \sin \chi_i \right)}{m_c + \sum_{i=1}^2 m_i \left(1 - \frac{3}{4} \cos^2 \chi_i \right)}$$

$$\ddot{\chi} = -\frac{3}{8l_i} \left(\ddot{x} \cos \chi_i + g \sin \chi_i + 2 \frac{\mu_i \dot{\chi}_i}{m_i l_i} \right)$$

这里, 杆的长度分别为 $l_1 = 1\text{m}$ 和 $l_2 = 0.1\text{m}$, 车的质量 $m_c = 1\text{kg}$, 杆的质量分别为 $m_1 = 0.1\text{kg}$ 和 $m_2 = 0.01\text{kg}$, $g = 9.81\text{m/s}^2$ 是重力常数。摩擦力使用系数 $\mu_c = 5 \cdot 10^{-4}\text{Ns/m}$ 和 $\mu_1 = \mu_2 = 2 \cdot 10^{-6}\text{Nms}$ 建模。允许的状态空间用车的位置 $x \in [-2.4\text{m}, 2.4\text{m}]$ 定义, 两个杆的角度都用 $\chi_i \in [-36^\circ, 36^\circ]$ 定义 ($i \in \{1, 2\}$)。在离开允许的状态空间时, 该阶段停止。每个时间步产生一个 $r_t = 1$ 的回报, 因此, 使阶段尽量长时是最优的。学习器可以每 0.02s 从范围 $[-50\text{N}, 50\text{N}]$ 中选择一个动作。

因为 CMA-ES 使用蒙特卡罗 roll-outs, 通过每 20s 重新设置环境, 任务成为显式的阶段性 [Heidrich-Meisner and Igel, 2008]。对于 Cacla, 这是不需要的, 但是为了使比较公平, 在 Cacla 中也做了。特征向量是 $\phi(s) = (x, \dot{x}, \chi_1, \dot{\chi}_1, \chi_2, \dot{\chi}_2)^T$ 。所有的阶段以 $\phi(s) = (0, 0, 1^\circ, 0, 0, 0)^T$ 开始。论文中的阻尼系数是 $\gamma = 1$ 。意思是说, 状态值是无界的。因此, 我们使用阻尼系数 $\gamma = 0.99$ 。实践中, 这对性能几乎没有差别。即使 Cacla 优化了折扣的累积回报, 我们仍然为

每阶段使用该回报作为性能的度量，CMA-ES 和 NAC 显式地优化了该回报。

用 CMA-ES 和 NAC 来训练一个线性控制器，因而，也用 Cacla 找到一个控制器。我们使用一个有偏的总是等于 1 的特征，所以我们在寻找一个参数向量 $\psi \in \mathbb{R}^7$ 。使用一个硬阈值，使得如果 actor 的输出大于 50N 或者小于 -50N，学习器分别输出 50N 或者 -50N。用一个具有 40 个隐藏节点、隐藏层使用 tanh 激活函数的多层感知机实现 critic。最初的控制

器用 -0.3 到 0.3 之间的均匀随机参数初始化。没有做其他的尝试来优化该参数的或者隐藏节点的数目的最初范围。

CMA-ES 的结果展示在图 7.4 中。[Heidrich-Meisner and Igel, 2008] 展示了 NAC 表现很差，因此，我们没有展示它的性能。如果 NAC 初始化为接近最优策略，那么它的性能会变得更好，在这种情况下，NAC 的平均性能在 3000 到 4000 个阶段之后，在每 1000 个阶段，达到了最优回报。然而，这当然将假设一个关于最优解的先验知识。CMA-ES 最好性能是一个中位回报，大约为 850。如图 7.4 所示，对于参数 σ 的值，除了 $\sigma=1$ ，性能是糟糕的。

我们用固定的步长大小 $\alpha = \beta = 10^{-3}$ 和高斯搜索 $\sigma = 5000$ ，只运行 Cacla 500 个阶段。粗略地调整后者的值，搜索如此高的原因是 Cacla 高效地学习一个 bang-bang 控制器：结果所得 actor 输出值远大于 50N 以及远低于 -50N。结果对于搜索参数的精确设置是非常健壮的。

我们也用 ϵ -贪婪搜索执行了 Cacla，其中，选择了概率为 ϵ 的均匀随机动作。使用 $\epsilon=0.1$ 和 $\epsilon=1$ ，后者表明了一个完全随机的策略。Cacla 的 ϵ -贪婪版本不去学习一个 bang-bang 控制器，因为该 actor 的目标总是在许可范围内。注意，当 $\epsilon=0.1$ 时，actor 只平均每 10 步更新一次。

表 7.2 展示了使用 Calca 的结果。展示了每阶段（最大为 1000）的平均回报以及成功率，后者是可以使杆保持平衡至少 20 秒的控制器的百分比。在线列展示了对于阶段 401 ~ 500 的 averages 的在线结果，包括搜索。离线列展示了平均离线结果，通过在 500 个阶段结束之后，不使用搜索来测试 actor。图 7.4 展示了平均性能，但是，在 $\sigma = 5000$ 以及 $\epsilon=0.1$ 时，Cacla 的在线和离线平均性能已经在 1000 上完全正确了。这可以从表 7.2 中看出来，因为在这些情况下，成功率超过了 50%。为了能够比较 Cacla 的不同的搜索技术，我们展示了平均性能。

表 7.2 每个片段的平均回报（均值）、均值（se）的标准误差和试验百分比，每个片段的回报等于 1000（成功），其中， $\alpha=\beta = 10^{-3}$ 。图中显示了训练片段 401 ~ 500（在线）的结果和贪婪策略 500 次训练（离线）的结果。动作噪声和搜索在正文中进行了解释。在 1000 次重复中取平均值

动作噪声	搜索	在线			离线		
		均值	均值的标准误差	成功率	均值	均值的标准误差	成功率
0	$\sigma = 5000$	946.3	6.7	92.3%	954.2	6.3	94.5%
	$\epsilon = 0.1$	807.6	9.6	59.0%	875.2	9.4	84.5%
	$\epsilon = 1$	29.2	0.0	0%	514.0	10.4	25.5%

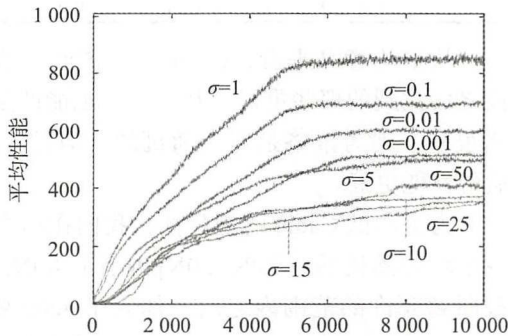


图 7.4 500 次重复实验中，每个片段的中位回报。x 轴表示片段数量。图来自 [Heidrich-Meisner and Igel, 2008 年]

(续)

动作噪声	搜索	在线			离线		
		均值	均值的标准误差	成功率	均值	均值的标准误差	成功率
[-20N,20N]	$\sigma = 5000$	944.6	6.9	92.5%	952.4	6.5	94.5%
	$\varepsilon = 0.1$	841.0	8.7	60.7%	909.5	8.1	87.4%
	$\varepsilon = 1$	28.7	0.0	0%	454.7	9.5	11.3%
[-40N,40N]	$\sigma = 5000$	936.0	7.4	91.9%	944.9	7.0	93.8%
	$\varepsilon = 0.1$	854.2	7.9	50.5%	932.6	6.7	86.7%
	$\varepsilon = 1$	27.6	0.0	0%	303.0	6.7	0%

从平均意义来看, Cacla 只用 500 个阶段找到的控制器明显比 CMA-ES 在 10 000 个阶段之后找到的那些更好。甚至 $\varepsilon=1$ 也能产生非常合理的贪婪策略。当然, 当 $\varepsilon=1$ 时, 在线性能很差, 因为策略是完全随机的。但是, 注意, 贪婪性能 514.0 比 CMA-ES 在 500 个阶段后的性能更好。

为了测试 Cacla 的健壮性, 我们在动作执行时加入噪声, 重新运行实验。在执行前, 将一个均匀随机的 $[-20N, 20N]$ 或者 $[-40N, 40N]$ 范围内的力加到了动作上。在将 actor 输出截断到允许的范围内之后, 加入了 actor 噪声, 算法不知道添加的噪声的量。例如, 假设 Cacla 的 actor 输出 $Ac(s_t) = 40$ 的一个动作。然后, 加入高斯搜索, 例如, 结果 $a_t=70$ 。动作没有在允许的范围 $[-50, 50]$ 内, 所以它被截断到 50。然后, 加入来自 $[-20, 20]$ 或者 $[-40, 40]$ 的均匀噪声。假设结果是 60。然后, 一个 60N 的力施加到了车上。如果结果时序差分是正的, 对于该状态, actor 的输出朝着 $a_t=70$ 进行更新, 所以, 算法既不知道截断, 也不知道施加到它的输出上的均匀噪声。

表 7.2 展示了包括动作噪声的结果。当使用高斯搜索时, Cacla 的性能几乎不受影响。在统计误差范围内, 性能略微下降, 虽然, 它看起来是一致的。有趣的是, 增加的噪声甚至在使用 $\varepsilon=1$ 的 ε -贪婪搜索时, 提高了 Cacla 的在线和离线性能。很明显, 增加的噪声导致了令人满意的额外的搜索。

实验表明, 相当简单的 Cacla 算法在解决一些连续强化学习问题时是非常有效的。其他
[241] 以前的工作显示, 自然梯度和进化算法通常需要几千个阶段才能在双杆中学习到一个好的策略, 而且, 在单杆任务 [Sehnke et al, 2010] 上也是如此。这里我们没有展示这些结果, 但是 Cacla 在单杆车杆上同样表现良好。当然, 这不表示 Cacla 对所有的连续 MDP 都是最好的选择。例如, 在一个部分可观察的 MDP 中, 进化方法直接在参数空间进行搜索可能更快地找到一个好的控制器, 虽然, 使用 Cacla 训练一个递归神经网络也是可能的, 例如, 实时递归学习 [Williams and Zipser, 1989] 或者反向传播 [Werbos, 2002, 2002]。此外, 对于 Cacla 的变体, 仍然不能保证收敛到一个最优甚至局部最优策略, 但是, 对于一些 actor-critic 算法 [Bhatnagar et al, 2009] 和直接策略搜索算法可以保证收敛到一个局部最优。

在这个具体的问题上, Cacla 远比 CMA-ES 表现得更好。原因是, CMA-ES 使用整个阶段来估计一个候选策略的适应度, 而且存储这些策略的整个种群。另一方面, Cacla 通过使用时序差分误差, 利用了问题的结构。这使得它快速更新它的 actor, 使得即使在前几个阶段学习成为可能。NAC 有另外的缺点, 须要相当多的样本来使得对 Fisher 信息矩阵的估计足够精确, 从而找到自然梯度方向。最后, 在值空间的平稳时期, Cacla 对 actor 的改进没有减慢。随着阶段变得更长, 值空间通常将表现出这样的平稳期, 使得 NAC 所用的梯度估计变得更加不可靠, 更新更小。因为 Cacla 直接在动作空间进行操作, 它没有这个问题, 它可

以使用一个固定的步长，每当时序差分为正时，朝着更好的动作方向移动。

最后一个注意事项，Cacla 的简单变体将可能在具有特殊类型噪声的问题中表现不好。例如，Cacla 可能试图朝着通常产生相当高的返回但是有时候产生非常低的返回的方向进行更新，使得在平均意义上，产生一个差的选择。可以通过存储回报函数的一个显式的近似来缓解这个问题，或者通过使用平均化的时序差分误差而不是随机误差。目前还没有对这些问题进行深入的研究。

7.5 总结

有很多种方法从具有连续空间的问题中找到好的策略。在显示逼近问题的一部分存在三种不同的常见方法：模型、策略的值或者策略本身。函数逼近可用于逼近这些函数，可以用基于梯度或无梯度方法更新。许多不同的强化学习算法来自这些技巧的组合。我们主要集中在价值函数和策略逼近方面，因为连续的 MDP 模型迅速变得难以解决，使这些显式近似不再那么有用。

[242]

我们讨论了几种方法来更新价值函数逼近器，包括时序差分算法，例如 TD 学习、Q 学习、GTD2 和 TDC。要更新策略逼近器，可用的方法包括：策略-梯度方法、actor-critic 算法和进化算法。因为这些方法为策略存储一种显式的近似，它们可以直接应用于动作空间也是连续的问题上。

在这些方法中，无梯度直接策略搜索算法在完全连续的问题中有最佳的收敛保证。然而，这些方法是低效的，因为它们使用完全的蒙特卡罗 roll-outs，并且不利用 MDP 的马尔可夫结构。actor-critic 方法存储显式估计策略和利用这种结构的 critic，例如，通过使用时序差分。这些方法有很大的潜力，尽管它们通常很难分析。

为了得到不同方法的优点，连续的 actor-critic 学习自动机 (Cacla) 算法 [van Hasselt and Wiering, 2007, 2009] 在一个双极平衡问题上与最新的黑箱优化和直接策略搜索方法进行了比较：协方差矩阵适应进化策略 (CMA-ES) [Hansen et al, 2003]。在前面的工作中，CMA-ES 优于其他方法，例如自然进化策略 (NES) [Wierstra et al, 2008 ;Sun et al, 2009] 和自然 actor-critic (NAC) 算法 [Peters and Schaal, 2008a]。我们的研究表明，Cacla 在更小数量的阶段中达到更好的策略。一个原因是 Cacla 是一种在线的 actor-critic 方法，而其他方法需要更多的样本来推断出策略要更新到的方向。换句话说，Cacla 能更有效地利用可用经验样本，尽管它可以很容易地扩展到更多的样本。

连续的 MDP 比有限的 MDP 有更小的总体收敛性保证。最近已经有一些研究工作来填补这一空白（例如，见 [Bertsekas, 2007 ;Szepesvári, 2010]，但仍然需要更多的分析。许多目前的方法在大问题计算方面是（部分地）启发式、样本效率低下或棘手的。然而，近年来，逐渐出现了理论保证和实用的通用算法，我们期望这一趋势将继续保持下去。在具有大的或连续域的通用问题中，有效地找到最优决策是在人工智能领域中最困难的问题之一，但它也是一个具有许多实际应用和影响的话题。

致谢。我想感谢 Peter Bosman 和匿名的审阅者提出的非常有用的意见。

参考文献

Akimoto, Y., Nagata, Y., Ono, I., Kobayashi, S.: Bidirectional Relation Between CMA Evolution Strategies and Natural Evolution Strategies. In: Schaefer, R., Cotta, C., Kołodziej, J.,

- Rudolph, G. (eds.) PPSN XI. LNCS, vol. 6238, pp. 154–163. Springer, Heidelberg (2010)
- Albus, J.S.: A theory of cerebellar function. *Mathematical Biosciences* 10, 25–61 (1971)
- Albus, J.S.: A new approach to manipulator control: The cerebellar model articulation controller (CMAC). In: *Dynamic Systems, Measurement and Control*, pp. 220–227 (1975)
- Amari, S.I.: Natural gradient works efficiently in learning. *Neural Computation* 10(2), 251–276 (1998)
- Anderson, C.W.: Learning to control an inverted pendulum using neural networks. *IEEE Control Systems Magazine* 9(3), 31–37 (1989)
- Antos, A., Munos, R., Szepesvári, C.: Fitted Q-iteration in continuous action-space MDPs. In: *Advances in Neural Information Processing Systems (NIPS-2007)*, vol. 20, pp. 9–16 (2008a)
- Antos, A., Szepesvári, C., Munos, R.: Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning* 71(1), 89–129 (2008b)
- Babuska, R.: *Fuzzy modeling for control*. Kluwer Academic Publishers (1998)
- Bäck, T.: *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford University Press, USA (1996)
- Bäck, T., Schwefel, H.P.: An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation* 1(1), 1–23 (1993)
- Baird, L.: Residual algorithms: Reinforcement learning with function approximation. In: *Prieditis, A., Russell, S. (eds.) Machine Learning: Proceedings of the Twelfth International Conference*, pp. 30–37. Morgan Kaufmann Publishers, San Francisco (1995)
- Baird, L.C., Klopff, A.H.: Reinforcement learning with high-dimensional, continuous actions. Tech. Rep. WL-TR-93-114, Wright Laboratory, Wright-Patterson Air Force Base, OH (1993)
- Bardi, M., Dolcetta, I.C.: *Optimal control and viscosity solutions of Hamilton–Jacobi–Bellman equations*. Springer, Heidelberg (1997)
- Barto, A.G., Sutton, R.S., Anderson, C.W.: Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics SMC-13*, 834–846 (1983)
- Baxter, J., Bartlett, P.L.: Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research* 15, 319–350 (2001)
- Beard, R., Saridis, G., Wen, J.: Approximate solutions to the time-invariant Hamilton–Jacobi–Bellman equation. *Journal of Optimization theory and Applications* 96(3), 589–626 (1998)
- Bellman, R.: *Dynamic Programming*. Princeton University Press (1957)
- Benbrahim, H., Franklin, J.A.: Biped dynamic walking using reinforcement learning. *Robotics and Autonomous Systems* 22(3–4), 283–302 (1997)
- Berenji, H.: Fuzzy Q-learning: a new approach for fuzzy dynamic programming. In: *Proceedings of the Third IEEE Conference on Fuzzy Systems, IEEE World Congress on Computational Intelligence*, pp. 486–491. IEEE (1994)
- Berenji, H., Khedkar, P.: Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks* 3(5), 724–740 (1992)
- Bertsekas, D.P.: *Dynamic Programming and Optimal Control*, vol. I. Athena Scientific (2005)
- Bertsekas, D.P.: *Dynamic Programming and Optimal Control*, vol. II. Athena Scientific (2007)
- Bertsekas, D.P., Tsitsiklis, J.N.: *Neuro-dynamic Programming*. Athena Scientific, Belmont (1996)
- Bertsekas, D.P., Borkar, V.S., Nedic, A.: Improved temporal difference methods with linear function approximation. In: *Handbook of Learning and Approximate Dynamic Programming*, pp. 235–260 (2004)
- Beyer, H., Schwefel, H.: Evolution strategies—a comprehensive introduction. *Natural Computing* 1(1), 3–52 (2002)
- Bhatnagar, S., Sutton, R.S., Ghavamzadeh, M., Lee, M.: Natural actor-critic algorithms. *Automatica* 45(11), 2471–2482 (2009)
- Bishop, C.M.: *Neural networks for pattern recognition*. Oxford University Press, USA (1995)

- Bishop, C.M.: Pattern recognition and machine learning. Springer, New York (2006)
- Bonardini, A.: Delayed reinforcement, fuzzy Q-learning and fuzzy logic controllers. In: Herrera, F., Verdegay, J.L. (eds.) Genetic Algorithms and Soft Computing. Studies in Fuzziness, vol. 8, pp. 447–466. Physica-Verlag, Berlin (1996)
- Boyan, J.A.: Technical update: Least-squares temporal difference learning. Machine Learning 49(2), 233–246 (2002)
- Bradtke, S.J., Barto, A.G.: Linear least-squares algorithms for temporal difference learning. Machine Learning 22, 33–57 (1996)
- Bryson, A., Ho, Y.: Applied Optimal Control. Blaisdell Publishing Co. (1969)
- Buşoniu, L., Ernst, D., De Schutter, B., Babuška, R.: Continuous-State Reinforcement Learning with Fuzzy Approximation. In: Tuyls, K., Nowe, A., Guessoum, Z., Kudenko, D. (eds.) ALAMAS 2005, ALAMAS 2006, and ALAMAS 2007. LNCS (LNAI), vol. 4865, pp. 27–43. Springer, Heidelberg (2008)
- Buşoniu, L., Babuška, R., De Schutter, B., Ernst, D.: Reinforcement Learning and Dynamic Programming Using Function Approximators. CRC Press, Boca Raton (2010)
- Coulom, R.: Reinforcement learning using neural networks, with applications to motor control. PhD thesis, Institut National Polytechnique de Grenoble (2002)
- Crites, R.H., Barto, A.G.: Improving elevator performance using reinforcement learning. In: Touretzky, D.S., Mozer, M.C., Hasselmo, M.E. (eds.) Advances in Neural Information Processing Systems, vol. 8, pp. 1017–1023. MIT Press, Cambridge (1996)
- Crites, R.H., Barto, A.G.: Elevator group control using multiple reinforcement learning agents. Machine Learning 33(2/3), 235–262 (1998)
- Davis, L.: Handbook of genetic algorithms. Arden Shakespeare (1991)
- Dayan, P.: The convergence of $TD(\lambda)$ for general lambda. Machine Learning 8, 341–362 (1992)
- Dayan, P., Sejnowski, T.: $TD(\lambda)$: Convergence with probability 1. Machine Learning 14, 295–301 (1994)
- Dearden, R., Friedman, N., Russell, S.: Bayesian Q-learning. In: Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence, pp. 761–768. American Association for Artificial Intelligence (1998)
- Dearden, R., Friedman, N., Andre, D.: Model based Bayesian exploration. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, pp. 150–159 (1999)
- Eiben, A.E., Smith, J.E.: Introduction to evolutionary computing. Springer, Heidelberg (2003)
- Ernst, D., Geurts, P., Wehenkel, L.: Tree-based batch mode reinforcement learning. Journal of Machine Learning Research 6(1), 503–556 (2005)
- Fisher, R.A.: On the mathematical foundations of theoretical statistics. Philosophical Transactions of the Royal Society of London Series A, Containing Papers of a Mathematical or Physical Character 222, 309–368 (1922)
- Fisher, R.A.: Statistical methods for research workers. Oliver & Boyd, Edinburgh (1925)
- Främling, K.: Replacing eligibility trace for action-value learning with function approximation. In: Proceedings of the 15th European Symposium on Artificial Neural Networks (ESANN-2007), pp. 313–318. d-side publishing (2007)
- Gaskett, C., Wettergreen, D., Zelinsky, A.: Q-learning in continuous state and action spaces. In: Advanced Topics in Artificial Intelligence, pp. 417–428 (1999)
- Geramifard, A., Bowling, M., Sutton, R.S.: Incremental least-squares temporal difference learning. In: Proceedings of the 21st National Conference on Artificial Intelligence, vol. 1, pp. 356–361. AAAI Press (2006)
- Geramifard, A., Bowling, M., Zinkevich, M., Sutton, R.: ilstd: Eligibility traces and convergence analysis. In: Advances in Neural Information Processing Systems, vol. 19, pp. 441–448 (2007)
- Glassmachers, T., Schaul, T., Yi, S., Wierstra, D., Schmidhuber, J.: Exponential natural evolution strategies. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, pp. 393–400. ACM (2010)
- Glorennec, P.: Fuzzy Q-learning and dynamical fuzzy Q-learning. In: Proceedings of the Third IEEE Conference on Fuzzy Systems, IEEE World Congress on Computational In-

- telligence, pp. 474–479. IEEE (1994)
- Glover, F., Kochenberger, G.: Handbook of metaheuristics. Springer, Heidelberg (2003)
- Gomez, F., Schmidhuber, J., Mäkeläinen, R.: Accelerated neural evolution through cooperatively coevolved synapses. *The Journal of Machine Learning Research* 9, 937–965 (2008)
- Gordon, G.J.: Stable function approximation in dynamic programming. In: Prieditis, A., Russell, S. (eds.) *Proceedings of the Twelfth International Conference on Machine Learning (ICML 1995)*, pp. 261–268. Morgan Kaufmann, San Francisco (1995)
- Gordon, G.J.: Approximate solutions to Markov decision processes. PhD thesis, Carnegie Mellon University (1999)
- Greensmith, E., Bartlett, P.L., Baxter, J.: Variance reduction techniques for gradient estimates in reinforcement learning. *The Journal of Machine Learning Research* 5, 1471–1530 (2004)
- Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation* 9(2), 159–195 (2001)
- Hansen, N., Müller, S.D., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation* 11(1), 1–18 (2003)
- Hansen, N., Auger, A., Ros, R., Finck, S., Pošik, P.: Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In: *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation, GECCO 2010*, pp. 1689–1696. ACM, New York (2010)
- Haykin, S.: *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR (1994)
- Heidrich-Meisner, V., Igel, C.: Evolution Strategies for Direct Policy Search. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) *PPSN 2008. LNCS, vol. 5199*, pp. 428–437. Springer, Heidelberg (2008)
- Holland, J.H.: Outline for a logical theory of adaptive systems. *Journal of the ACM (JACM)* 9(3), 297–314 (1962)
- Holland, J.H.: *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor (1975)
- Howard, R.A.: *Dynamic programming and Markov processes*. MIT Press (1960)
- Huyer, W., Neumaier, A.: SNOBFIT—stable noisy optimization by branch and fit. *ACM Transactions on Mathematical Software (TOMS)* 35(2), 1–25 (2008)
- Jiang, F., Berry, H., Schoenauer, M.: Supervised and Evolutionary Learning of Echo State Networks. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) *PPSN 2008. LNCS, vol. 5199*, pp. 215–224. Springer, Heidelberg (2008)
- Jouffe, L.: Fuzzy inference system learning by reinforcement methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 28(3), 338–355 (1998)
- Kakade, S.: A natural policy gradient. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) *Advances in Neural Information Processing Systems 14 (NIPS-2001)*, pp. 1531–1538. MIT Press (2001)
- Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: *Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, vol. 4*, pp. 1942–1948 (1995)
- Kirkpatrick, S.: Optimization by simulated annealing: Quantitative studies. *Journal of Statistical Physics* 34(5), 975–986 (1984)
- Klir, G., Yuan, B.: *Fuzzy sets and fuzzy logic: theory and applications*. Prentice Hall PTR, Upper Saddle River (1995)
- Konda, V.: Actor-critic algorithms. PhD thesis, Massachusetts Institute of Technology (2002)
- Konda, V.R., Borkar, V.: Actor-critic type learning algorithms for Markov decision processes. *SIAM Journal on Control and Optimization* 38(1), 94–123 (1999)
- Konda, V.R., Tsitsiklis, J.N.: Actor-critic algorithms. *SIAM Journal on Control and Optimization* 42(4), 1143–1166 (2003)
- Kullback, S.: *Statistics and Information Theory*. J. Wiley and Sons, New York (1959)
- Kullback, S., Leibler, R.A.: On information and sufficiency. *Annals of Mathematical Statistics* 22, 79–86 (1951)
- Lagoudakis, M., Parr, R.: Least-squares policy iteration. *The Journal of Machine Learning Research* 4, 1107–1149 (2003)

- Lin, C., Lee, C.: Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems. *IEEE Transactions on Fuzzy Systems* 2(1), 46–63 (1994)
- Lin, C.S., Kim, H.: CMAC-based adaptive critic self-learning control. *IEEE Transactions on Neural Networks* 2(5), 530–533 (1991)
- Lin, L.: Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning* 8(3), 293–321 (1992)
- Lin, L.J.: Reinforcement learning for robots using neural networks. PhD thesis, Carnegie Mellon University, Pittsburgh (1993)
- Littman, M.L., Szepesvári, C.: A generalized reinforcement-learning model: Convergence and applications. In: Saitta, L. (ed.) *Proceedings of the 13th International Conference on Machine Learning (ICML 1996)*, pp. 310–318. Morgan Kaufmann, Bari (1996)
- Maei, H.R., Sutton, R.S.: GQ (λ): A general gradient algorithm for temporal-difference prediction learning with eligibility traces. In: *Proceedings of the Third Conference On Artificial General Intelligence (AGI-2010)*, pp. 91–96. Atlantis Press, Lugano (2010)
- Maei, H.R., Szepesvári, C., Bhatnagar, S., Precup, D., Silver, D., Sutton, R.: Convergent temporal-difference learning with arbitrary smooth function approximation. In: *Advances in Neural Information Processing Systems 22 (NIPS-2009)* (2009)
- Maei, H.R., Szepesvári, C., Bhatnagar, S., Sutton, R.S.: Toward off-policy learning control with function approximation. In: *Proceedings of the 27th Annual International Conference on Machine Learning (ICML-2010)*. ACM, New York (2010)
- Maillard, O.A., Munos, R., Lazaric, A., Ghavamzadeh, M.: Finite sample analysis of Bellman residual minimization. In: *Asian Conference on Machine Learning, ACML-2010* (2010)
- Mitchell, T.M.: *Machine learning*. McGraw Hill, New York (1996)
- Moriarty, D.E., Miikkulainen, R.: Efficient reinforcement learning through symbiotic evolution. *Machine Learning* 22, 11–32 (1996)
- Moriarty, D.E., Schultz, A.C., Grefenstette, J.J.: Evolutionary algorithms for reinforcement learning. *Journal of Artificial Intelligence Research* 11, 241–276 (1999)
- Murray, J.J., Cox, C.J., Lendaris, G.G., Sacks, R.: Adaptive dynamic programming. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 32(2), 140–153 (2002)
- Narendra, K.S., Thathachar, M.A.L.: Learning automata - a survey. *IEEE Transactions on Systems, Man, and Cybernetics* 4, 323–334 (1974)
- Narendra, K.S., Thathachar, M.A.L.: *Learning automata: an introduction*. Prentice-Hall, Inc., Upper Saddle River (1989)
- Nedić, A., Bertsekas, D.P.: Least squares policy evaluation algorithms with linear function approximation. *Discrete Event Dynamic Systems* 13(1-2), 79–110 (2003)
- Neyman, J., Pearson, E.S.: On the use and interpretation of certain test criteria for purposes of statistical inference part i. *Biometrika* 20(1), 175–240 (1928)
- Ng, A.Y., Parr, R., Koller, D.: Policy search via density estimation. In: Solla, S.A., Leen, T.K., Müller, K.R. (eds.) *Advances in Neural Information Processing Systems*, vol. 13, pp. 1022–1028. The MIT Press (1999)
- Nguyen-Tuong, D., Peters, J.: Model learning for robot control: a survey. *Cognitive Processing*, 1–22 (2011)
- Ormoneit, D., Sen, Š.: Kernel-based reinforcement learning. *Machine Learning* 49(2), 161–178 (2002)
- Pazis, J., Lagoudakis, M.G.: Binary action search for learning continuous-action control policies. In: *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 793–800. ACM (2009)
- Peng, J.: Efficient dynamic programming-based learning for control. PhD thesis, Northeastern University (1993)
- Peters, J., Schaal, S.: Natural actor-critic. *Neurocomputing* 71(7-9), 1180–1190 (2008a)
- Peters, J., Schaal, S.: Reinforcement learning of motor skills with policy gradients. *Neural Networks* 21(4), 682–697 (2008b)
- Peters, J., Vijayakumar, S., Schaal, S.: Reinforcement learning for humanoid robotics. In: *IEEE-RAS International Conference on Humanoid Robots (Humanoids 2003)*. IEEE Press (2003)

- Poupart, P., Vlassis, N., Hoey, J., Regan, K.: An analytic solution to discrete Bayesian reinforcement learning. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 697–704. ACM (2006)
- Powell, M.: UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming* 92(3), 555–582 (2002)
- Powell, M.: The NEWUOA software for unconstrained optimization without derivatives. In: *Large-Scale Nonlinear Optimization*, pp. 255–297 (2006)
- Powell, W.B.: *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley-Blackwell (2007)
- Precup, D., Sutton, R.S.: Off-policy temporal-difference learning with function approximation. In: *Machine Learning: Proceedings of the Eighteenth International Conference (ICML 2001)*, pp. 417–424. Morgan Kaufmann, Williams College (2001)
- Precup, D., Sutton, R.S., Singh, S.P.: Eligibility traces for off-policy policy evaluation. In: *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pp. 766–773. Morgan Kaufmann, Stanford University, Stanford, CA (2000)
- Prokhorov, D.V., Wunsch, D.C.: Adaptive critic designs. *IEEE Transactions on Neural Networks* 8(5), 997–1007 (2002)
- Puterman, M.L.: *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York (1994)
- Puterman, M.L., Shin, M.C.: Modified policy iteration algorithms for discounted Markov decision problems. *Management Science* 24(11), 1127–1137 (1978)
- Rao, C.R., Poti, S.J.: On locally most powerful tests when alternatives are one sided. *Sankhyā: The Indian Journal of Statistics*, 439–439 (1946)
- Rechenberg, I.: *Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Fromman-Holzboog (1971)
- Riedmiller, M.: Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) *ECML 2005. LNCS (LNAI)*, vol. 3720, pp. 317–328. Springer, Heidelberg (2005)
- Ripley, B.D.: *Pattern recognition and neural networks*. Cambridge University Press (2008)
- Rubinstein, R.: The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability* 1(2), 127–190 (1999)
- Rubinstein, R., Kroese, D.: *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*. Springer-Verlag New York Inc. (2004)
- Rückstieß, T., Sehnke, F., Schaul, T., Wierstra, D., Sun, Y., Schmidhuber, J.: Exploring parameter space in reinforcement learning. *Paladyn* 1(1), 14–24 (2010)
- Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: *Parallel Distributed Processing*, vol. 1, pp. 318–362. MIT Press (1986)
- Rummery, G.A., Niranjan, M.: On-line Q-learning using connectionist systems. Tech. Rep. CUED/F-INFENG-TR 166, Cambridge University, UK (1994)
- Santamaria, J.C., Sutton, R.S., Ram, A.: Experiments with reinforcement learning in problems with continuous state and action spaces. *Adaptive Behavior* 6(2), 163–217 (1997)
- Scherrer, B.: Should one compute the temporal difference fix point or minimize the Bellman residual? The unified oblique projection view. In: Fürnkranz, J., Joachims, T. (eds.) *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*, pp. 959–966. Omnipress (2010)
- Schwefel, H.P.: *Numerische Optimierung von Computer-Modellen*. Interdisciplinary Systems Research, vol. 26. Birkhäuser, Basel (1977)
- Sehnke, F., Osendorfer, C., Rückstieß, T., Graves, A., Peters, J., Schmidhuber, J.: Parameter-exploring policy gradients. *Neural Networks* 23(4), 551–559 (2010)
- Singh, S.P., Sutton, R.S.: Reinforcement learning with replacing eligibility traces. *Machine Learning* 22, 123–158 (1996)
- Spaan, M., Vlassis, N.: Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research* 24(1), 195–220 (2005)
- Stanley, K.O., Miikkulainen, R.: Efficient reinforcement learning through evolving neural

- network topologies. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002), pp. 569–577. Morgan Kaufmann, San Francisco (2002)
- Strehl, A.L., Li, L., Wiewiora, E., Langford, J., Littman, M.L.: PAC model-free reinforcement learning. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 881–888. ACM (2006)
- Strens, M.: A Bayesian framework for reinforcement learning. In: Proceedings of the Seventeenth International Conference on Machine Learning, p. 950. Morgan Kaufmann Publishers Inc. (2000)
- Sun, Y., Wierstra, D., Schaul, T., Schmidhuber, J.: Efficient natural evolution strategies. In: Proceedings of the 11th Annual conference on Genetic and Evolutionary Computation (GECCO-2009), pp. 539–546. ACM (2009)
- Sutton, R.S.: Temporal credit assignment in reinforcement learning. PhD thesis, University of Massachusetts, Dept. of Comp. and Inf. Sci. (1984)
- Sutton, R.S.: Learning to predict by the methods of temporal differences. *Machine Learning* 3, 9–44 (1988)
- Sutton, R.S.: Generalization in reinforcement learning: Successful examples using sparse coarse coding. In: Touretzky, D.S., Mozer, M.C., Hasselmo, M.E. (eds.) *Advances in Neural Information Processing Systems*, vol. 8, pp. 1038–1045. MIT Press, Cambridge (1996)
- Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. The MIT press, Cambridge (1998)
- Sutton, R.S., McAllester, D., Singh, S., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: *Advances in Neural Information Processing Systems 13 (NIPS-2000)*, vol. 12, pp. 1057–1063 (2000)
- Sutton, R.S., Szepesvári, C., Maei, H.R.: A convergent $O(n)$ algorithm for off-policy temporal-difference learning with linear function approximation. In: *Advances in Neural Information Processing Systems 21 (NIPS-2008)*, vol. 21, pp. 1609–1616 (2008)
- Sutton, R.S., Maei, H.R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C., Wiewiora, E.: Fast gradient-descent methods for temporal-difference learning with linear function approximation. In: *Proceedings of the 26th Annual International Conference on Machine Learning (ICML 2009)*, pp. 993–1000. ACM (2009)
- Szepesvári, C.: Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 4(1), 1–103 (2010)
- Szepesvári, C., Smart, W.D.: Interpolation-based Q-learning. In: *Proceedings of the Twenty-First International Conference on Machine Learning (ICML 2004)*, p. 100. ACM (2004)
- Szita, I., Lőrincz, A.: Learning tetris using the noisy cross-entropy method. *Neural Computation* 18(12), 2936–2941 (2006)
- Taylor, M.E., Whiteson, S., Stone, P.: Comparing evolutionary and temporal difference methods in a reinforcement learning domain. In: *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, p. 1328. ACM (2006)
- Tesauro, G.: Practical issues in temporal difference learning. In: Lippman, D.S., Moody, J.E., Touretzky, D.S. (eds.) *Advances in Neural Information Processing Systems*, vol. 4, pp. 259–266. Morgan Kaufmann, San Mateo (1992)
- Tesauro, G.: TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation* 6(2), 215–219 (1994)
- Tesauro, G.J.: Temporal difference learning and TD-Gammon. *Communications of the ACM* 38, 58–68 (1995)
- Thrun, S., Schwartz, A.: Issues in using function approximation for reinforcement learning. In: Mozer, M., Smolensky, P., Touretzky, D., Elman, J., Weigend, A. (eds.) *Proceedings of the 1993 Connectionist Models Summer School*. Lawrence Erlbaum, Hillsdale (1993)
- Touzet, C.F.: Neural reinforcement learning for behaviour synthesis. *Robotics and Autonomous Systems* 22(3/4), 251–281 (1997)
- Tsitsiklis, J.N., Van Roy, B.: An analysis of temporal-difference learning with function approximation. Tech. Rep. LIDS-P-2322, MIT Laboratory for Information and Decision Systems, Cambridge, MA (1996)
- Tsitsiklis, J.N., Van Roy, B.: An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control* 42(5), 674–690 (1997)

- van Hasselt, H.P.: Double Q-Learning. In: *Advances in Neural Information Processing Systems*, vol. 23. The MIT Press (2010)
- van Hasselt, H.P.: *Insights in reinforcement learning*. PhD thesis, Utrecht University (2011)
- van Hasselt, H.P., Wiering, M.A.: Reinforcement learning in continuous action spaces. In: *Proceedings of the IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL-2007)*, pp. 272–279 (2007)
- van Hasselt, H.P., Wiering, M.A.: Using continuous action spaces to solve discrete problems. In: *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2009)*, pp. 1149–1156 (2009)
- van Seijen, H., van Hasselt, H.P., Whiteson, S., Wiering, M.A.: A theoretical and empirical analysis of Expected Sarsa. In: *Proceedings of the IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pp. 177–184 (2009)
- Vapnik, V.N.: *The nature of statistical learning theory*. Springer, Heidelberg (1995)
- Vrabie, D., Pastravanu, O., Abu-Khalaf, M., Lewis, F.: Adaptive optimal control for continuous-time linear systems based on policy iteration. *Automatica* 45(2), 477–484 (2009)
- Wang, F.Y., Zhang, H., Liu, D.: Adaptive dynamic programming: An introduction. *IEEE Computational Intelligence Magazine* 4(2), 39–47 (2009)
- Watkins, C.J.C.H.: *Learning from delayed rewards*. PhD thesis, King's College, Cambridge, England (1989)
- Watkins, C.J.C.H., Dayan, P.: Q-learning. *Machine Learning* 8, 279–292 (1992)
- Werbos, P.J.: *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. PhD thesis, Harvard University (1974)
- Werbos, P.J.: Advanced forecasting methods for global crisis warning and models of intelligence. In: *General Systems*, vol. XXII, pp. 25–38 (1977)
- Werbos, P.J.: Backpropagation and neurocontrol: A review and prospectus. In: *IEEE/INNS International Joint Conference on Neural Networks*, Washington, D.C., vol. 1, pp. 209–216 (1989a)
- Werbos, P.J.: Neural networks for control and system identification. In: *Proceedings of IEEE/CDC*, Tampa, Florida (1989b)
- Werbos, P.J.: Consistency of HDP applied to a simple reinforcement learning problem. *Neural Networks* 2, 179–189 (1990)
- Werbos, P.J.: Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE* 78(10), 1550–1560 (2002)
- Whiteson, S., Stone, P.: Evolutionary function approximation for reinforcement learning. *Journal of Machine Learning Research* 7, 877–917 (2006)
- Whitley, D., Dominic, S., Das, R., Anderson, C.W.: Genetic reinforcement learning for neurocontrol problems. *Machine Learning* 13(2), 259–284 (1993)
- Wieland, A.P.: Evolving neural network controllers for unstable systems. In: *International Joint Conference on Neural Networks*, vol. 2, pp. 667–673. IEEE, New York (1991)
- Wiering, M.A., van Hasselt, H.P.: The QV family compared to other reinforcement learning algorithms. In: *Proceedings of the IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pp. 101–108 (2009)
- Wierstra, D., Schaul, T., Peters, J., Schmidhuber, J.: Natural evolution strategies. In: *IEEE Congress on Evolutionary Computation (CEC-2008)*, pp. 3381–3387. IEEE (2008)
- Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8, 229–256 (1992)
- Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. *Neural Computation* 1(2), 270–280 (1989)
- Wilson, D.R., Martinez, T.R.: The general inefficiency of batch training for gradient descent learning. *Neural Networks* 16(10), 1429–1451 (2003)
- Zadeh, L.: Fuzzy sets. *Information and Control* 8(3), 338–353 (1965)
- Zhou, C., Meng, Q.: Dynamic balance of a biped robot using fuzzy reinforcement learning agents. *Fuzzy Sets and Systems* 134(1), 169–187 (2003)

综述：求解一阶逻辑马尔可夫决策过程

Martijn van Otterlo

摘要

在本章中，我们综述了马尔可夫决策过程（Markov Decision Processes, MDP）的表示和技术、强化学习和动态规划。这样的关系世界在规划领域（planning domain）、游戏领域、现实世界的室内景色和更多场景下随处可见，关系表示（relational representation）允许以明确的方式捕获对象和关系的表达性的和自然的数据结构，在对象和关系上启用一般化，但也有类似的不同的对象数量的问题。[van Otterlo, 2009b] 最近实地调查了这一领域，这里描述了大部分的主要方法。我们讨论无模型的（包括基于数值的和基于策略的）和基于模型的动态规划技术。我们也将讨论一些其他的问题，包括模型和等级，并在本章的末尾讨论最新的进展和将来的方向。

读者需要着重理解以下概念：知识表达、动态规划、关系马尔可夫决策过程（Relational Markov Decision Processes, RMDP）、马尔可夫决策过程、决策-理论回归（Decision-Theoretic Regression, DTR）、一阶决策-理论回归、象征性动态规划（Symbolic Dynamic Programming, SDP）、一阶拟合线性优化（First-Order Approximate Linear Programming, FOALP）技巧、一阶拟合策略迭代（First-Order Approximate Policy Iteration, FOAPI）算法、基于关系实例的（Relational Instance Based, RIB）回归方法、基于抽象化和一般化的策略迭代、关系型强化学习、部分观察的马尔可夫决策过程（Partially Observable Markov Decision Processes, POMDP）等。

8.1 关系世界中的顺序决策简介

作为人类，我们谈论并认为世界是由物体和物体之间的关系构成的。书、桌子和房子、房子里面的桌子、桌子上的书等。“我们配备了感应偏压，学会把世界分割成物体，学习这些物体之间的相互作用，将各种计算模块应用于表示这些对象” [Baum, 2004, p. 173]。对于智能的学习器来说，这不应该是不同的。“实际上，很难想象一个真正的智能的学习器不会根据对象及其属性与其他对象的关系来构想世界” [Kaelbling et al, 2001]。进而，这种表示法（representation）是非常紧凑和有效的，“而用物体和简单交互描述世界是一个非常压缩的描述” [Baum, 2004, p. 168]。

最近十年间，强化学习（Reinforcement Learning, RL）的新分支出现了：这些新的分支通过利用知识表达（Knowledge Representation, KR）的一阶逻辑构成的智能学习器处理对象和关系，通过高效的强化学习算法在复杂和非确定的世界中处理决策理论学习（decision-theoretic learning）。研究关系强化学习这个领域 [van Otterlo, 2009b] 需要灵感、代表性方法和不同领域的算法（见图 8.2a），而这些算法包括强化学习 [Sutton and Barto, 1998]、基于逻辑

辑的人工智能 [Minker, 2000]、知识表达 [Brachman and Levesque, 2004]、规划 (planning) [Russell and Norvig, 2003]、概率逻辑机器学习 [De Raedt, 2008] 和认知结构 [Langley, 2006]。在这些领域, 对象和关系的使用是广泛的, 假设强化学习能够运用于这些领域。例如, 认知学习器能够工作, 或者在任务中强化学习需要表达已获得的知识, 研究人员必须研究强化学习算法如何与表现的形式化进行交互, 例如一阶逻辑 (first-order logic)。关系型强化学习 (Relational Reinforcement Learning, RRL) 为强化学习提供一种新的代表性范式 (representational paradigm), 关系型强化学习能比最新的建议方法更紧凑更高效地解决许多问题, 此外, 关系型强化学习能解决以前难以解决的问题。关系型强化学习还为在算法中注入额外的背景知识提供了新的机会。

在本章中, 我们介绍了在强化学习中作为新的代表范式的关系型强化学习。首先介绍强化学习中的概括要素并简述历史发展。8.2 节中, 我们为马尔可夫决策过程引入了关系表示。在 8.3 节和 8.4 节之中, 我们综述了基于模型的和无模型的算法。在 8.5 节中, 我们综述了其他方法, 包括层次结构 (hierarchies) 和模型学习 (model learning)。在 8.6 节中, 我们描述了最新的进展。在 8.7 节中, 我们总结并提出了未来的方向。

8.1.1 马尔可夫决策过程: 代表性和可扩展性

给定马尔可夫决策过程 $M = \langle S, A, T, R \rangle$, 通常的目标是计算价值函数 (即 V 或者 Q), 或直接计算策略 π 。为了实现这些目标, 存在大量的算法, 这些算法或多或少都属于广义策略迭代 (GPI) 的范畴 [Sutton and Barto, 1998]。广义策略迭代包含以下循环:

- 1) 策略评估, 计算 V^π 。
- 2) 策略提高, 从 V^π 中计算提高的策略 π 。

广义策略迭代是为不透明状态和动作而制定的, 而不考虑表示法和一般化的使用。

我们制定了一个更加一般化的概念——基于抽象化和一般化的策略迭代 (Policy Iteration using Abstraction and Generalization Techniques, PIAGET), 在基于抽象化和一般化的策略迭代概念中, 我们必须显式地使用表示 [van Otterlo, 2009b], 见图 8.1。基于抽象化和一般化的策略迭代概念区分了广义策略迭代与表示相互作用的四种方式。PIAGET-0 第一次产生一个抽象层的紧凑的表示, 例如通过最小化模型 (model minimization) 或者提取特征 (feature extraction)。PIAGET-1 算法和 PIAGET-2 算法与广义策略迭代相近, 因为这些算法假设一个固定的表示形式, 并学习通过抽象层 (PIAGET-1) 学习马尔可夫决策过程相关的函数 (Q 、 V), 或者通过抽象层 (PIAGET-2) 学习参数 (例如神经网络的权重) PIAGET-3 方法更加一般化, 当求解马尔可夫决策过程的时候, PIAGET-3 方法采用了抽象层 (abstraction level)。

因为关系型强化学习首先是一个具有代表性的升级版本的强化学习, 在马尔可夫决策过程的算法中, 我们要看概念的概括或抽象。我们可以区分五种马尔可夫决策过程的常用类型: 状态空间抽象、分解马尔可夫决策过程表示、价值函数、策略、层次分解 (hierarchical decomposition)。关系型强化学习侧重于在对象和关系方面表示马尔可夫决策过程算法, 使用逻辑归纳法获得五种类型中的任何一种, 可能利用建议表示

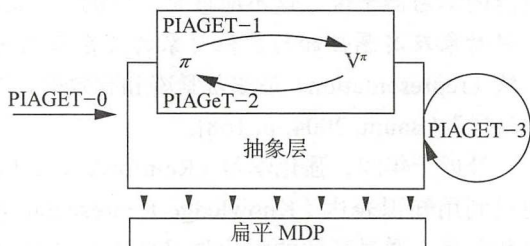


图 8.1 基于抽象化和一般化的策略循环 (PIAGET)

法 (propositional representation) 得到的结果。

概括与表达密切相关。换言之, 仅仅能学到能够表达的部分。尽管有许多表达知识 (representing knowledge) 的方法, 包括规则、神经网络、实体关系模型 (entity-relationship model)、一阶逻辑等等, 表达的种类只有几种:

- **原子的 (atomic)**。大多数算法和证明的描述都使用所谓的原子状态表示法 (atomic state representation), 状态空间 S 是一个离散的集合, 包括离散的状态 s_1, \dots, s_N , 而 A 是动作的集合。没有使用抽象或一般化, 所有数值的计算和存储都发生在状态之间。
- **建议的 (propositional)**。这是马尔可夫决策过程最常见的表示形式, 见 [Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998; Boutilier et al, 1999]。每个状态由属性-数值对 (attribute-value pair) 组成的向量组成, 每一个属性 (特征、随机变量), 表示域的可测量数量 (measurable quantity)。特征可以是二值的 (binary), 也可以是实数值的 (real-valued)。通常, 动作集合 A 仍然是一个离散的集合。建议编码 (propositional encoding) 允许使用诸如决策树或者支持向量机 (Support Vector Machine, SVM) 来表征价值函数和策略, 从而对状态空间的多个部分进行一般化。
- **指示的 (deictic)**。有些工作使用指示表示法 (如 [Finney et al, 2002]), 而其中的建议编码是指对象。例如, 特征的值“在我手上的物体的颜色”能够表达一个特定的物体 (specific object), 而不是显式地命名该物体。指示表示法和面向对象的表示法 (object-oriented representations) [Diuk et al, 2008] 依然是提议中的, 但是填补了显式的关系形式 (explicit relational formalisms) 之间的差距。
- **相关性 (relational)**。关系表示法是一种表达性知识表达 (KR) 格式, 对象和对象之间的关系可以用显式的方式来表达, 这将在下一节讨论。

每个表征类 (representational class) 都有自己的概括和抽象方法。从那个方面来说, 原子状态 (atomic state) 不能够提供更多, 但是对于建议表示法来说是可能的。一般化利用了问题域固有的结构。可以通过两种不同的方式发现和利用这种结构。人们不仅可以利用结构化的表示法, 使解决方案更加紧凑, 而且可以在算法中利用结构表述, 使强化学习算法更加高效。

[255]

8.1.2 简短的历史和与其他领域的联系

本章所描述的研究是至少三个领域的自然发展。如图 8.2 所示, 核心的元素是一阶或关系型马尔可夫决策过程。

- **强化学习**。在 20 世纪 90 年代中期以前, 多次尝试和错误学习 (trial-and-error learning)、优化控制 (optimal control)、动态规划和价值函数逼近 (value function approximation) 的技巧发展起来了。[Sutton and Barto, 1998] 的书标志着一个新时代的开始, 在这个时代里, 发展了更为复杂的算法和表达方式, 关于逼近和收敛形成了更多的理论。在千禧年之交, 关系型强化学习发展成一个新的领域, 远远超出了建议表示法的范畴。[Džeroski et al, 1998] 开发了相关领域的 Q 学习的第一个版本。[Boutilier et al, 2001] 报告了相关领域的第一个一阶逻辑的数值迭代算法 (value iteration algorithm)。在强化学习领域共同发起了一个新的代表方向。

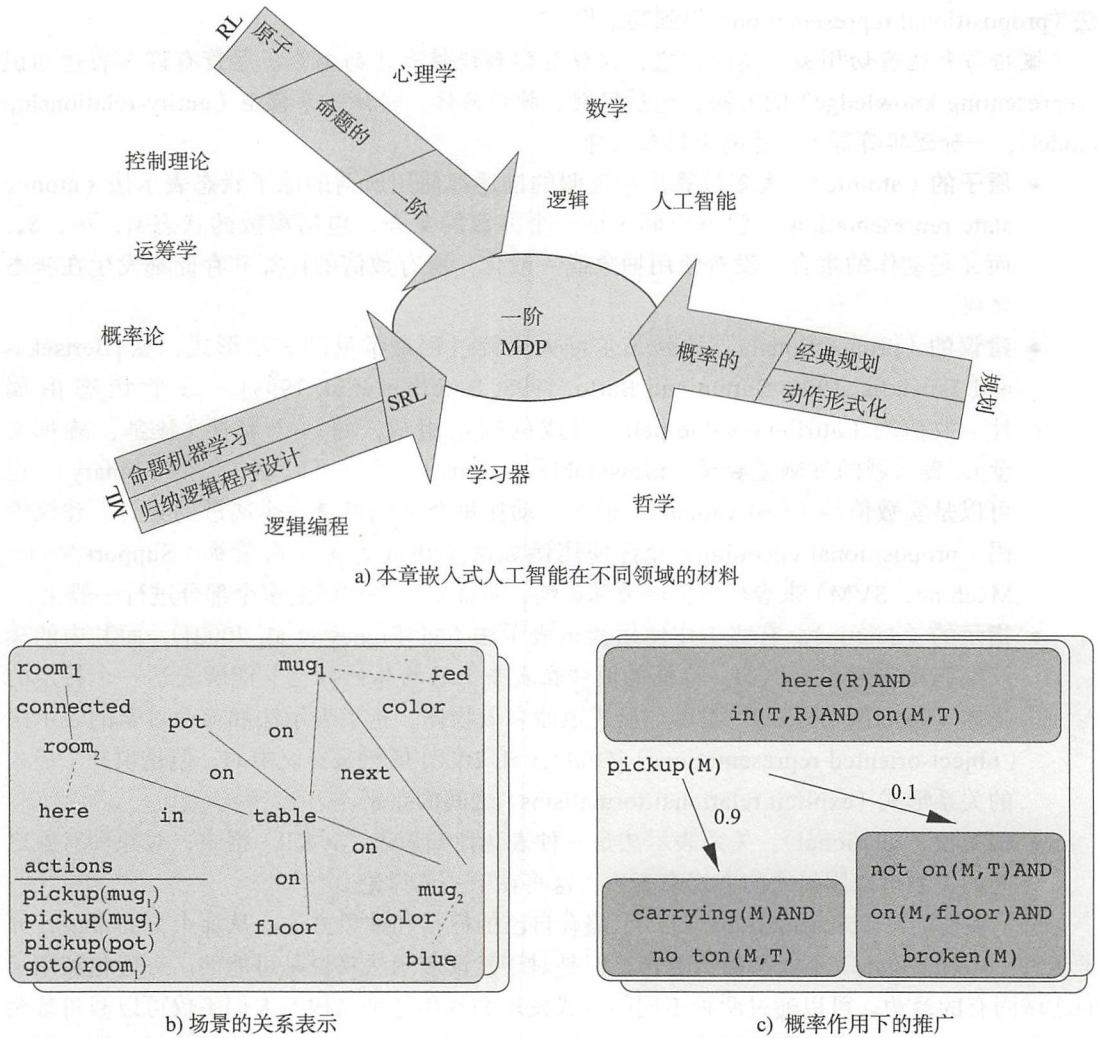


图 8.2 a) 本章嵌入式人工智能在不同领域的材料；b) 场景的关系表示；c) 概率作用下的推广

- **机器学习。**机器学习 (Machine Learning, ML) 的强化学习是一个子领域，更早的使用关系表示法。归纳逻辑程序设计 (Inductive Logic Programming, ILP) 在从数据中学习逻辑概念 (logical concept) 方面有着悠久的历史 [Bergadano and Gunetti, 1995]。前仍有大量机器学习研究使用纯粹的提议表示法，侧重于概率方面 [Alpaydin, 2004]。然而，在统计关系学习 (Statistical Relational Learning, SRL) 领域，最近十年的逻辑和概率方法正在合并 [De Raedt, 2008]。关系型强化学习本身可以看作是一个额外的步骤，将功能框架 (utility framework) 加入统计关系学习中。
- **动作语言。**规划和认知结构 (cognitive architecture) 是老的人工智能话题，但在关系型强化学习之前，很少有方法能够有效地处理决策理论概念。一阶动作语言包括 STRIPS[Fikes and Nilsson, 1971]、情景演算 [Reiter, 2001] 等，主要用于确定的、基于目标的规划上下文，这对于认知结构来说也是一样的。许多关系型强化学习是基于现有的动作的形式主义 (action formalisms)。因此，扩展其适应性以求解决决策理论问题。

8.2 用面向对象和关系扩展马尔可夫决策过程

这里我们介绍关系表示和逻辑归纳 (logical generalization), 以及如何运用关系表示和逻辑归纳对关系型 (或一阶逻辑) 马尔可夫决策过程进行正则化叙述。我们的描述将比较非正式, 详见 [van Otterlo, 2009b], 逻辑归纳的综述见 [Brachman and Levesque, 2004]。

8.2.1 关系表示与逻辑归纳

如图 8.2b 所示, 对象和关系需要用显式的方式进行形式化表述。对象之间的关系是实线 (room1, room2), 例如相互连接的关系 (table, floor)。有些关系 (即虚线) 仅仅表示对象的性质, 并可以用颜色来表示 (mug₁, red) 或用一元关系表示 red(mug₁)。动作以类似的方式表示, 例如带参数的渲染 Pickup(mug₁)、跳转语句 goto(room1) 等。

注意, 场景的建议表征会很麻烦。对象之间的每个关系都应该用二进制特征 (binary feature) 表示。为了用一个特征向量 (feature vector) 来表示一个状态, 所有的对象和关系必须先固定、排序并产生特征。例如, 应该包含 on_mug₁_table=1 和 on_mug₂_table=1, 但也包含无用特征, 如 on_table_table=0 和 on_room₁_mug₂=0。因为必须包含许多不相关的关系, 这将引起大爆炸, 如对象或关系的数量变化也不太灵活。关系表示可以自然地处理这些问题。 [257]

概括对象的一个重要方法是: 使用可以代表不同对象的变量 (用大写字母表示)。例如, 在图 8.2c 中, 动作规范利用其来拾取由 m 表示的对象。其指定对象应该在 t 表示的某个对象上, 在相同的当前位置 r 中, 并且 m 可以是 mug, 例如: mug₁ 或 mug₂。底部两个矩形显示了两种不同的动作结果, 受概率分布支配。

如图 8.3a 所示, 在许多人工领域之外, 积木世界 (Blocks World, BW) 问题可能是以下领域最著名的问题 [Slaney and Thiébaux, 2001], 包括知识表达 (KR) 和规划算法、关系型强化学习算法等 [Russell and Norvig, 2003; Schmid, 2001; Brachman and Levesque, 2004]。对于通用人工智能系统 (general purpose AI systems) 来说, 这是一个计算复杂的问题, 这个系统拥有相对简单的形式, 并支持有价值的、系统的和可支付的实验。积木世界问题通常用于关系型强化学习, 我们在这里使用关系型强化学习来解释重要概念。

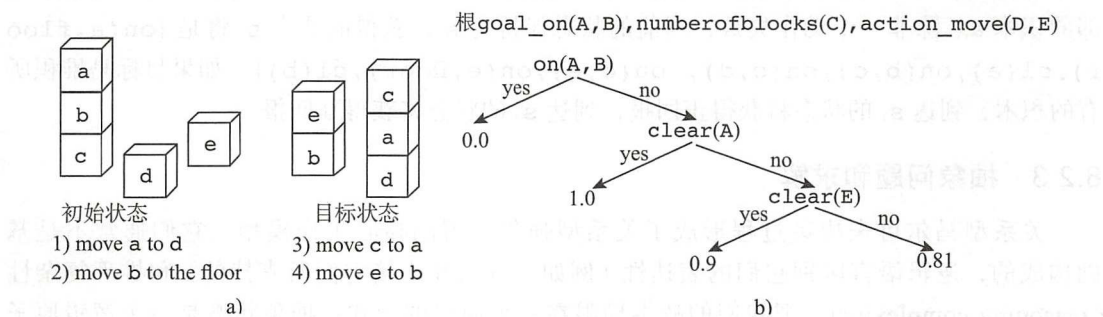


图 8.3 a) 积木规划问题与最优方案; b) 逻辑 Q 树

更一般地, 一个关系表示由一个域对象 (或常数) C 和一组关系 $\{p/\alpha\}$ (或谓词) 组成, 而且每个可以有几个参数 (即数量 α)。真实原子 (ground atom) 是应用于 C 常数的关系, 例如 on (a, b)。推而广之, 除了使用域 C 和套谓词 P , 通常的逻辑语言还包含变量、量词和连词。句子 (即公式) $\mathbb{Z} \equiv \exists X, Y \text{ on } (X, Y) \wedge \text{clear}(X)$ 中, \mathbb{Z} 代表所有的状态存在

(由量词 \exists 表示)任意两个对象(或块)的 x 和 y (通过连接 \wedge 或逻辑运算符AND表示),且 x 是明确的。在图8.3a中,在初始状态 x 只能是 a ,但在目标状态中可以指 c 或 e 。注意,因为包含变量, z 不是真实的。

逻辑抽象可以从数据中学习,这通常通过ILP算法[Bergadano and Gunetti, 1995],或者SRL算法[De Raedt, 2008]完成。这些方法的细节描述不在本章的范围之内。在这里,我们可以把这些算法看作是通过搜索大量逻辑抽象空间的算法,类似于建议树和规则学习器[Alpaydin, 2004]。空间的结构是由逻辑语言表达的。在SRL算法的例子中,附加的问题是学习参数,例如概率。

8.2.2 关系型马尔可夫决策过程

我们首先需要用关系形式来表示问题的所有方面,包括用关系形式表示的马尔可夫决策过程算法,而这个算法是基于对象领域 D 和一系列的预测值 P 。

定义 8.2.1 假设 $P = \{p_1/\alpha_1, \dots, p_n/\alpha_n\}$ 是具有一系列元数的预测值, $C = \{c_1, \dots, c_k\}$ 是一系列的常数,那么, $A' = \{a_1/\alpha_1, \dots, a_m/\alpha_m\}$ 是一系列具有元数的动作。假设 S' 为由 P 和 C 构成的所有基态原子的集合, A 是所有由 A' 和 C 构成的基态原子的集合。

258

关系型马尔可夫决策过程(RMDP)是一个元组 $M = \langle S, A, T, R \rangle$,其中, S 是 $2^{S'}$ 的一个子集和, $T :: S \times A \times S \rightarrow [0,1]$ 是一个概率转换函数 $R :: S \times A \times S \rightarrow \mathbb{R}$ 是奖励函数。

因此,每一个状态由在该状态中为真的基础关系原子集表达,动作称为基础关系原子,转换和奖励函数像往常一样定义,但是现在超越了基础关系状态(ground relational state)和基础关系动作(ground relational action)。这就是说,与建议表示法不同,状态是关系原子的非排序集合,原子的数目也不是固定的。

示例 8.2.1 使用基于预测集合 $P = \{\text{on}/2, \text{cl}/1\}$ 的五积木问题(five-block world)的例子,常数集合 $C = \{a, b, c, d, e, \text{floor}\}$ 和动作集合 $A' = \{\text{move}/2\}$,我们可以定义积木问题,而积木问题包含5个积木和 $|S| = 501$ 个合法状态。 T 能定义为符合标准的积木问题移动算子,也许包括一些失败的可能性。 R 能设置成正的状态,满足一些目标条件。

一个具体的状态的 s_1 是 $\{\text{on}(a, b), \text{on}(b, c), \text{on}(c, d), \text{on}(d, e), \text{on}(e, \text{floor}), \text{cl}(a)\}$,其中,所有的积木都堆积起来了。动作执行 $\text{move}(a, \text{floor})$ 动作,并移动了最顶部的积木 a 。除非一个动作失败,当前的状态保持为 s_1 ,获得的状态 s_2 将是 $\{\text{on}(a, \text{floor}), \text{cl}(a), \text{on}(b, c), \text{on}(c, d), \text{on}(d, e), \text{on}(e, \text{floor}), \text{cl}(b)\}$ 。如果目标是堆积所有的积木,到达 s_1 的状态将获得正回报,到达 s_2 的状态将获得0回报。

8.2.3 抽象问题和求解

关系型马尔可夫决策过程形成了关系型强化学习的核心工作模型。它们通常不是基础构成的,逻辑语言区别它们的表达性(例如,量化和连接它们所支持的)和推理复杂性(reasoning complexity)。把我们的故事局限在一个简单的形式:抽象状态是一个逻辑原子(logical atom)的复合体 $Z \equiv Z_1 \wedge \dots \wedge Z_m$,包含了很多变量。复合体(conjunction)是隐式的量化存在,即抽象状态 $Z_1 \equiv \text{on}(X, Y)$ 视为 $\exists X \exists Y Z_1$,其中包含两个块:记作 x 块和 y 块且 x 块在 y 块上面。为了简洁起见,我们将从现在开始省略限定符(quantifier)和连接符(connector)。如果找到一个 Z 中的替代变量使得所有替代原子出现在 Z 中,一个抽象状态 Z 将模拟一个基础状态 z (ground state)。例如,我们能够以 a 和 b 替代 x 和 y 。因此,一



一个抽象状态可以一般化为关系型马尔可夫决策过程包含的一系列状态。即，这个抽象状态是一个聚和状态 (aggregate state)。

抽象状态以 θ -包含 (θ -subsumption) 排序。如果一个抽象状态 \mathbb{Z}_1 包含另外一个抽象状态 \mathbb{Z}_2 ，那么抽象状态 \mathbb{Z}_1 比抽象状态 \mathbb{Z}_2 更加广泛。一个领域理论 (domain theory) 支持推理 (reasoning) 和约束处理 (constraint handling)，来检查状态是否合法 (与域的底层语义相关) 和可扩展 (即基于领域理论得出新的事实)。如果 $\text{under}(b,a)$ 为真，对于积木问题来说，从 $\text{on}(a,b)$ 演化过来是可能的。另外，这也排除了从关系原子中容易生成但在域中不可能的状态，例如，使用规则 $\text{on}(X,X) \rightarrow \text{false}$ 的 $\text{on}(a,a)$ 。

定义 8.2.1 的转换函数经常通过抽象的动作来定义。它们的定义依赖于特定的动作逻辑 (action logic) 和许多不同的形式。一个动作导致了抽象状态之间的映射，即从一个状态集映射到另一个状态集。一个抽象动作通过带有一系列确定的动作结果的概率分布定义了一个概率动作算子 (probabilistic action operator)。基于概率 STRIPS 算法 [Hanks and McDermott, 1994]，抽象动作的定义如下：

$$\begin{aligned} \text{cl}(X), \text{cl}(Y), \text{on}(X,Z), \xrightarrow{0.9 : \text{move}(X,Y)} & \text{on}(X,Y), \text{cl}(X), \text{cl}(Z), \\ X \neq Y, Y \neq Z, X \neq Z & X \neq Y, Y \neq Z, X \neq Z \\ \\ \text{cl}(X), \text{cl}(Y), \text{on}(X,Z), \xrightarrow{0.1 : \text{move}(X,Y)} & \text{cl}(X), \text{cl}(Y), \text{on}(X,Z), \\ X \neq Y, Y \neq Z, X \neq Z & X \neq Y, Y \neq Z, X \neq Z \end{aligned} \quad (8.1)$$

在 0.9 的概率下，移动块 x 和块 y 。在 0.1 的概率下，我们不用改变状态。对于一个从前到后的动作规则来说，后续的状态中去掉了 $\text{pre } \theta$ ，并加入 $\text{post } \theta$ 。应用于 $\mathbb{Z} \equiv \text{cl}(a), \text{cl}(b), \text{on}(a,c)$ ，这个动作告诉我们 $\text{move}(a,b)$ 将以 0.9 的概率导致 $\mathbb{Z}' \equiv \text{on}(a,b), c|(a), c|(c)$ 将以 0.1 的概率导致原地不动。一个动作将在状态-动作-状态对定义一个概率映射 (probabilistic mapping)。

抽象 (状态) 奖励函数很容易与抽象状态一起定义。一个统一的对 R 的定义是集合 $\{\mathbb{Z}_i, r | i = 1 \dots N\}$ ，其中每个 \mathbb{Z}_i 是一个抽象状态， r 是一个数值。对于关系型马尔可夫决策过程的每一个关系状态 \mathbb{Z} ，我们能够定义将要给出 \mathbb{Z} 的奖励和将一般化 \mathbb{Z} 的抽象状态 (即将归入 \mathbb{Z})。如果奖励对于状态来说是唯一的， R 形成了一个关系型马尔可夫决策过程的全部状态空间的划分。

整体来说，通过设置一个域 C 和一个关系集合 P 、一个抽象动作定义和一个抽象奖励函数，可以将关系型马尔可夫决策过程定义成简要的形式。通过只改变领域 C 可以获得关系型马尔可夫决策过程的变体，该算法包含多个对象的集合，即关系型马尔可夫决策过程的集合，详见 [van Otterlo, 2009b]。

对于解元素，例如策略和价值函数，我们可以使用相似的表示法。实际上，尽管价值函数与奖励函数是有区别的，我们可以用相同的表示法来表示一组具有值的抽象状态。然而，更加简洁的表示法也是可能的。为了表示状态-动作价值函数，[Džeroski et al, 1998] 采用一棵关系型决策树 (relational decision tree)，即将状态-动作空间紧凑地划分成具有相同数值的区域。图 8.3b 描述了一棵树，这棵树的根节点 (root node) 表示动作 $\text{move}(D,E)$ 。所有的状态-动作对希望在 E 上移动 D 块，其中块 A 不在块 B 上且 A 为 clear 。需要注意的是，与建议树相反是，节点测试 (node test) 可以共享变量。

策略从状态映射到动作，且能够经常使用关系型决策列表 (relational decision lists) 来表示它们 [Mooney and Califf, 1995]。例如，为积木问题考虑以下抽象策略，这个策略对 $\text{on}(a,b)$ 保存的达到状态进行优化编码：




```

r1: move(X,floor) ← onTop(X,b)
r2: move(Y,floor) ← onTop(Y,a)
r3: move(a,b)      ← clear(a),clear(b)
r4: noop           ← on(a,b)

```

其中 `noop` 表示什么都不做。规则 (rule) 可以从顶到底读取：给予一个状态，应用抽象状态的第一个规则产生了最佳动作。

无模型的或者基于模型的关系型强化学习需要实现产生表示法和控制学习。关系型马尔可夫决策过程的起始点 (starting point) 以及与此问题相对应的相关的价值函数 V 、价值函数 Q 和策略 π ，学习的最终目标是一个优化的抽象策略 $\tilde{\pi}$ 。

$$\text{RMDPM}\langle S, A, T, R \rangle \xrightarrow{\text{控制学习} + \text{表示学习}} \tilde{\pi}^*: S \rightarrow A \quad (8.2)$$

因为基础关系型马尔可夫决策过程中学习策略是不可能的，可以采取不同的策略来发现 $\tilde{\pi}^*$ 。首先构造抽象价值函数，然后从抽象价值函数推导出策略。从最佳的地面轨迹 (optimal ground traces) 来学习策略。关系型马尔可夫决策过程的关系抽象导致了许多结构学习任务 (structural learning task)。在下一节中，我们综述了不同的方式

8.3 基于模型的解决方案

基于模型的算法依赖于以下假设：关系型马尔可夫决策过程的完整模型是可用的，这些模型可以用于在动态规划算法中计算价值函数和策略。特色的解决模式 (characteristic solution pattern) 是以下纯粹的演绎步骤 (deductive step) 的序列：

$$\begin{array}{ccccccc}
 \tilde{V}^0 \equiv \mathcal{R} & \xrightarrow{D} & \tilde{V}^1 & \xrightarrow{D} & \tilde{V}^2 & \xrightarrow{D} & \dots \xrightarrow{D} \tilde{V}^k & \xrightarrow{D} & \tilde{V}^{k+1} & \xrightarrow{D} & \dots \\
 \downarrow D & & \downarrow D & & \downarrow D & & \downarrow D & & \downarrow D & & \\
 \tilde{\pi}^0 & & \tilde{\pi}^1 & & \tilde{\pi}^2 & & \tilde{\pi}^k & & \tilde{\pi}^{k+1} & &
 \end{array} \quad (8.3)$$

这些问题的初始规范 (initial specification) 能够视为逻辑理论，起始的奖励函数为 R ，用于初始化价值函数 \tilde{V}^0 。每一个随后的抽象价值函数 \tilde{V}^{k+1} 是通过演绎 (**D**) 从价值函数的 \tilde{V}^k 得到的。每一个价值函数 \tilde{V}^k 可以演绎得到一个策略 $\tilde{\pi}^k$ 。在本节的结尾，我们将简短地讨论附加的基于取样的方法。基于关系模型的求解算法通常使用贝尔曼最优函数。通过将这些求解算法转换为更新规则，它们将大量利用知识表达能力来探索解决方案和算法中的结构。这将导致几个关系型版本的传统算法，包括值迭代 (Value Iteration, VI)。

[261]

8.3.1 贝尔曼备份的结构

让我们来看看如下贝尔曼更新规则，通常同时应用于一个循环的所有状态 $s \in S$ ：

$$V^{k+1}(s) = (B^* V^k)(s) = R(s) + \gamma \max_a \sum_{s' \in S} T(s, a, s') V^k(s') \quad (8.4)$$

当在值迭代中使用的时候，基于备份算子 B^* ，贝尔曼更新规则将价值函数 V^k 的 k 步扩展到 $(k+1)$ 步进价值函数 V^{k+1} 。当深入更多的细节，我们可以区别以下操作，这些操作同时计算状态 $s \in S$ 的新数值 $V^{k+1}(s)$ ：

1) 匹配 / 重叠 (matching/Overlap)：我们观察状态 s' 可以通过一些动作来读写，其中我们知道的是 $V^k(s')$ 。



2) 回归 (regression): s 的值是通过备份状态的值来计算的, 而这些状态能够通过从 s 执行动作 a 进行访问。相等的, 如果我们知道 $V^k(s')$ 的值, 我们知道哪些状态 s 满足 $T(s, a, s') > 0$ 的条件, 我们能够通过后推理的方式从 s' 到 s 推断 Q 值 $Q(s, a) = \gamma \cdot T(s, a, s') \cdot V^k(s')$ 。

3) 混合 (combination): 既然动作 a 拥有多个概率的输出, 即转换到其他状态, 当运用到状态 s 的时候, 通过累加的方式合并部分的 Q 值, 获得 Q 值 $Q(s, a) = \sum_{s'} \in s T(s, a, s') V^k(s')$ 。之后, 奖励 $R(s)$ 能够加合起来。

4) 最大化 (maximization): 在每一个循环之中, 最高的状态值是需要。因此, 在可以运用的动作中最大化最后值 $V^{k+1}(s)$ 。

也能研究相似的算法, 例如策略迭代。

内涵的动态规划

现在, 传统的 VI 为每一个状态独立地计算值, 即使许多状态将相同的转换模式 (transition pattern) 共享到其他状态。结构表示法 (structured representation) 和关系表示法能够用于避免冗余, 能够同时为状态计算值。重温公式 (8.1) 定义的抽象动作。通过规则行事, 公式 (8.1) 紧凑地指定许多转换概率规则。相似的方式, 抽象价值函数能够运用简单的抽象状态简洁地表达一系列状态的值。内涵动态规划 (Intensional Dynamic Programming, IDP) [van Otterlo, 2009a] 显式地运用马尔可夫决策过程中知识表达形式, 并为结构动态规划提供了一个统一的框架。内涵动态规划定义为相互独立的表示法, 内涵动态规划能够以原子表示、建议表示或关系表示的方式实例化。内涵动态规划的核心包含通过表示法相关的组成部分来表达四个刚刚提到的计算: 重叠、回归、混合和最大化。这四个计算统称为决策-理论回归 (DTR)。内涵动态规划的简单应用实例是基于集合的动态规划, 其中价值函数将被分散集合表示且动态规划应用基于集合的备份。文献中的一些其他算法能够看成内涵动态规划的应用实例。

262

内涵动态规划的第一个实现是基于解释的强化学习 (Explanation-Based Reinforcement Learning, EBRL), 见 [Dietterich and Flann, 1997]。这个工作的灵感来自于贝尔曼备份与在基于解释的学习 (Explanation-Based Learning, EBL) 中的基于解释的一般化 (Explanation-Based Generalization, EBG), 见 [Minton et al, 1989]。运用的表示法包含建议的规则和基于区域的表示法 (即对于 2D 网格世界)。[Boutilier et al, 2000] 介绍结构化的值迭代算法 (SVI) 和策略迭代算法 (SPI), 这些算法是基于价值函数、策略和动态贝叶斯网络 (dynamic Bayesian network) 来表达转换函数的建议树, 见 [Boutilier et al, 2000]。后来, 这些技巧扩展到简洁的代数决策图 (Algebraic Decision Diagram, ADD)。内涵动态规划的第三个实现是连续状态空间的方形划分 [Feng et al, 2004]。这里, 所谓的矩形分段常数 (Rectangular Piecewise-Constant, RPWC) 函数通过 KD 树 (沿 K 轴分裂超平面) 进行存储, 便于高效的利用。内涵动态规划技巧在概念上来说容易扩展到部分观察的马尔可夫决策过程 [Boutilier and Poole, 1996]。让我们现在开始讨论关系型设置。

8.3.2 确切的基于模型的算法

关系型马尔可夫决策过程的确切的基于模型的算法实现了基于特定逻辑形式的决策-理论回归的四个组成部分。公式 (8.1) 的起始点是一个抽象的动作。假设我们的奖励函数 R , 其初始价值函数 V^0 是 $\{<\text{on}(a, b), \text{on}(c, d), 10>, <\text{true}, 0>\}$ 。现在, 在决策-理论回



归的第一步，我们能够使用动作规范（action specification）去发现抽象状态去备份 R 中的数值。既然价值函数不是依据个别状态而特定的，我们能够使用回归去发现动作的条件，目标是终止在抽象状态。让我们考虑动作的概率为 0.9 的第一个结果在价值函数 V^0 的运用。基于特定逻辑形式的决策 - 理论回归的一个组成部分匹配的目的是检查两个状态之间是否有重叠。也就是说，我们希望考虑被动作效应（the action effect）和我们观察的价值函数模拟的状态。如图 8.4 所示，可以证明：因为变量的使用（这里忽略约束），计算这个重叠有四个方法。

让我们研究图中的第一个可能性， (x,y) 与 (a,b) 区配，也就是说，我们考虑 a 在 b 上的动作。匹配产生了两个替代： x/a 和 y/b ，这两个替代导致了匹配的状态 $Z' \equiv \text{on}(a,b), \text{cl}(a), \text{cl}(Z)$ “加上” 动作没有导致的部分 $\text{on}(c,d)$ 。现在，回归步骤（regression step）是通过动作反向推理（reasoning backward），并发现为了得到 Z' 而采取的动作 $\text{move}(a,b)$ 的可能的抽象状态。这是一个直接的计算（straightforward computation），并导致 $Z \equiv \text{on}(a,Z), \text{cl}(b)$ “加上” $\text{on}(c,d)$ 。现在，根据决策 - 理论回归算法，我们可以计算出一个部分的 Q 值： $Q(Z, \text{move}(a,b)) = \gamma \cdot T(Z', \text{move}(a,b), \cdot) \cdot V^k(Z') = \gamma \cdot 0.9 \cdot 10$ 。

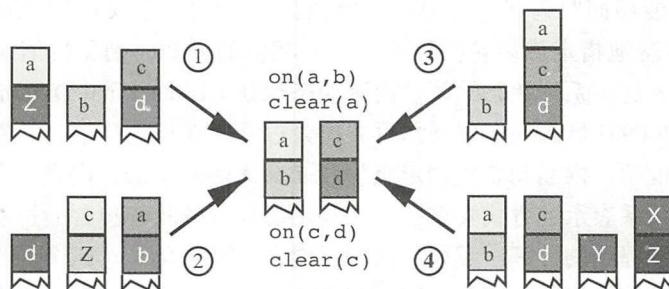


图 8.4 积木世界中的回归示例。所有在中心状态的四个配置是可能的“预状态”，通过一个标准的移动动作（move）来回归中心状态。这两个剩余的配置是 a 块防止在 b (1) 的情况，或者 c 积木防止在 d (2) 上的情况。这些配置中的 z 积木可以是一个积木或者平面（floor）。在顶部右边配置 (3) 积木放在 c 积木上，现在放在 b 上。第四种配置 (4) 是非常有趣的，因为这项配置假设，块 a, b, c, d 实际上移动了，但是有其他的积木移动了，即积木 x 和 y 。

所有的动作做这些步骤，价值函数 V^k 包含的所有动作的结果和所有的抽象状态将导致表示部分 Q 值的一系列的抽象状态 - 动作对 (Z, a, Q) 。在状态出现在部分 Q 函数的例子中，部分 Q 值与 Q 函数的混合是通过计算重叠实现的。让我们做出如下假设：对于 $Z^2 \equiv \text{on}(a, Z), \text{cl}(b), \text{on}(c,d), \text{on}(e,f)$ 来说，我们计算另一个部分 Q 值 $Q(Z^2, \text{move}(a, b)) = 3$ ，目的是以 0.1 的概率计算当前动作的第二个结果。现在，在重叠的状态 $Z^3 \equiv \wedge Z^2 \equiv \text{on}(a, Z), \text{cl}(b), \text{on}(c,d), \text{on}(e,f)$ ，我们知道在状态 Z^3 做动作 $\text{move}(a,b)$ ，其期望值为 $10+3$ 。对于所有可能的动作输出的混合来说，部分 Q 值对所有动作必须执行混合。

决策 - 理论回归的最后一个部分是最大化。在我们的例子中，这是非常简单的，因为抽象状态的自然归并秩序（natural subsumption order）做了大多数工作。可以对 Q 函数 $\{Z, A, Q\}$ 排序并对任意状态 - 动作对采取其包含的第一个规则。对于任意 Q 函数。基于效率方面的利益，若有一个更高价值的规则，可以去除 Q 函数包含的规则。对于其他的形式来说，最大化步骤有时候能够包含一个额外的推理工作用于最大化不同的变量替代物（variable



substitution)。整个过程概括为以下算法，称为一阶决策-理论回归 (First-Order Decision-Theoretic Regression, FOOTR):

Require: an abstract value function V^n

- 1: **for each** action type $A(X)$ **do**
- 2: compute $Q_{V^k}^{A(X)} = R \oplus [\gamma \otimes \oplus_j (\text{prob}(A(X)) \otimes \text{Regr}(V^k, A_j(X)))]$
- 3: $Q_{V^k}^A = \text{obj-max}(Q_{V^k}^{A(X)})$
- 4: $V^{k+1} = \max_A Q_{V^k}^A$

其中, \otimes 和 \oplus 表示抽象-值对 (abstraction-value pair) 的笛卡儿积 (Cartesian products) 的乘和加。Regr 表示回归算子且 obj-max 表示 Q 函数的最大化。

264

现在, 对于确切的一阶内涵动态规划算法已经在文献中出现了 4 个版本。第一个方法是基于情况微积分语言 (situation calculus language) 的象征性动态规划 (SDP) 方式 [Reiter, 2001]。第二个方法, FOVI [Holldobler and Skvortsova, 2004] 是基于流的微积分 [Thielscher, 1998]。第三个方法是 ReBel [Kersting et al, 2004]。最近的手段是基于一阶决策图 [Groote and Tveretina, 2003], 以及本章称为 FODD 的算法 [Wang et al, 2008a]。

象征性动态规划是第一个系统, 但它不存在一个实际的实现。一个重要的理由是象征性动态规划算法中的表达逻辑, 而这个逻辑是一阶决策-理论回归中的所有操作都变得计算上非常昂贵。尽管在表达上比较难懂, 其他的三个实现方式在计算上是合理的。图 8.5 显示了对于有 10 个积木的问题来说计算出的最优价值函数的目标 $\text{on}(a, b)$, 这个目标意味着六千万个基本状态。在相邻状态上的 ReBel 和 FOVI 操作之间只有很少交流, 其中 FODD 算法使用非常紧凑的数据结构。这就意味着在保持表示较小的情况下要付出更高的代价。就像其建议副本 (SVI) 一样, FODD 算法中的一阶决策-理论回归过程在整个价值函数上操作。其中, ReBel 算法、FOVI 算法和象征性动态规划算法在独立的抽象层工作。所有格式的转换函数都是基于逻辑的; ReBel 算法的概率 STRIPS 规则、FODD 算法的决策图、象征性动态规划算法和 FOVI 算法使用底层的流畅和情景演算的动作规范。图 8.6 显示出象征性动态规划算法和 FODD 算法使用的表示法的一些例子。注意, 象征性动态规划算法中的一个简单价值函数是非常精确的, 也是复杂的。FODD 算法的决策图是高度紧凑的, 但严格地说, 比象征性动态规划算法的状态描述缺乏表达能力。在象征性动态规划算法和 FOVI 算法中, 回归是内在的主要推理方法。但是, 对于 ReBel 算法和 FODD 算法来说, 发明了专门的过程。

所有的这四种方法在一阶域执行内涵动态规划算法, 没有首先接地的域, 从而直接在抽象层面计算解决方案。在一个更加广泛的层次, 一阶决策-理论回归能够看成在决策-理论价值之上执行一阶推理的手段, 即, 作为一类决策-理论逻辑。可以这样说, 所有四种方法通过一阶决策-理论回归, 使用动作定义和领域理论作为一组公理, (也许是无限的状态空间中) 演绎出最优状态。

下面描述四个系统的一些扩展。例如, 策略抽取和列表的使用 [van Otterlo, 2009b]、基于搜索的探索和有效的包含测试 [Karabaev et al, 2006]、策略迭代 [Wang and Khardon, 2007]、一阶马尔可夫决策过程的因素分解和附加奖励模型 [Sanner and Boutilier, 2007]、统一认证的目标 [Sanner and Boutilier, 2006] 等。



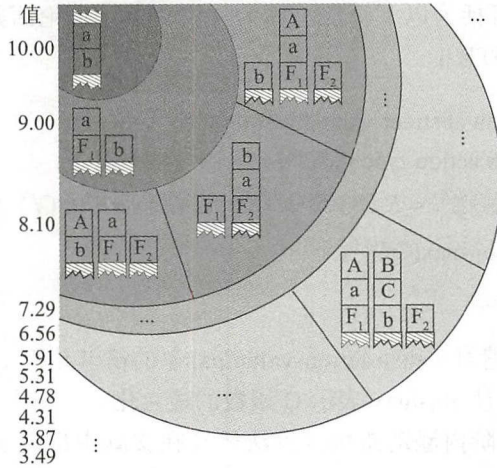


图 8.5 积木世界的抽象价值函数。10 个循环以后在 $\text{on}(a, b)$ 上的部分抽象价值函数。这包含超过一百条规则。 F^i 可以是一个积木或者地板积木。状态离开目标值 0.0 有 10 步之遥

265

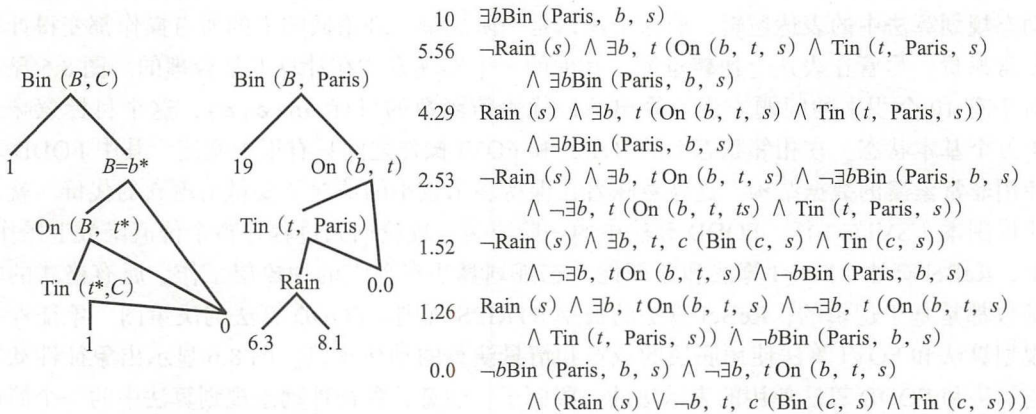


图 8.6 确切的一阶内涵动态规划的例子。我们实验章节中的例子都使有了逻辑域。左边的两幅图是一阶决策图 [Wang et al, 2007]。在左上角，转换函数（即真值图（truth value diagram））是为 $\text{Bin}(B, C)$ 描述的。右边的图描述了价值函数 \mathbb{V}^1 ，可以证明其与 $\mathbb{Q}_{\text{unload}}^1$ 是相等的。右边的等式代表了逻辑域的最终值分割 [Boutilier et al, 2001]

8.3.3 基于近似模型的算法

因为确切的最优价值函数的计算和存储是复杂的（在一些无限规模的领域，也许是无法定义的），有关关系型马尔可夫决策过程的基于近似模型的算法的一些工作已经开展起来。第一类的方法开始于一阶决策 - 理论回归然后再逼近，而第二类的方法使用了其他的方法，例如取样或者计划。

一阶近似线性规划技术（FOALP）[Sanner and Boutilier, 2005] 扩展了象征性动态规划，转化成一个近似值迭代算法 [Schuurmans and Patrascu, 2001]。除了确切的表达完全的价值函数，因为计算上困难，价值函数将是巨大的和细粒度的。与抽象状态对比，作者使用一个



固定集合的基函数。也就是说，一个价值函数能够表示成 k 个一阶的基函数的加权的和。其中，每一个基函数包含几个提供状态空间的一阶抽象（即划分）的公式。基函数的线性组合的备份是每一个基函数的一阶决策 - 理论回归的线性组合。与确定解不同，确定解的价值函数能够在规模上成倍增长且大量的努力用于简化逻辑，这个基于特征的方式为每一个状态只寻找好的权重（weights）。与一阶近似线性规划技术相关的是，一阶逼近策略迭代算法 [Sanner and Boutilier, 2006] 是因式分解的马尔可夫决策过程算法的一阶泛化 [Guestrin et al, 2003b]。就像一阶拟合线性优化技巧方法那样，使用了相同的基函数对价值函数进行分解。另外，还有显式的策略表达。算法在两个阶段之间循环。首先，计算出当前策略的价值函数，即，基函数的权重通过线性规划（LP）计算出来。第二个阶段从价值函数中计算策略。如果策略在连续的逼近中保持稳定，那么算法将获得收敛。收敛的策略的损失上界和下届是从因式分解马尔可夫决策过程中直接产生的。Guestrin 开发的 PRM 方法也为策略的质量提供上界和下界。然而，这些都是 PAC 上界和下界。这些上界和下界假设区域的概率与规模成指数级下降。在策略质量上，一阶近似线性规划技术的上下界对于所有的区域来说都是相等的。一阶近似线性规划技术已经应用于因式分解 MDP，并应用于 SysAdmin 区域 [Sanner and Boutilier, 2007]。一阶近似线性规划技术和一阶近似策略迭代都进入了国际计划竞赛（International Planning Competition, IPPC）的概率部分。在例如一阶近似线性规划技术相似的 AVI 框架中 [Wu and Givan, 2007] 描述了产生一阶特征的技巧。一个简单的 ILP 算法应用了特征空间中的柱搜索，ILP 算法的指导原则是：每一个特征与理想贝尔曼残差是交互的情况。

266

一个不同的方法是象征性动态规划的逼近 [Gretton and Thiébaux, 2004a, 2004b]。这个方法使用了象征性动态规划算法相同的基本设置，但是只是部分计算了一阶决策 - 理论回归过程。通过使用从目标状态的多步经典回归，计算出来许多结构用来代表抽象状态。组合步骤和最大化步骤没有完成。但是，回归产生的结构在高阶归纳树学习 ALKEMY 中，用作假设语言 [Lloyd, 2003] 以归纳表达价值函数的树。

第二类近似算法在抽象层不工作，正如一阶决策 - 理论回归及其扩展，但是在产生解的过程中使用取样和一般化。也就是说，首先产生一个解，或者更多的基本情况，包括计划、价值函数、策略，然后使用归纳概括的方法来获得一般化解。[Lecoeuche, 2001] 开创了此类求解方法，使用了关系型马尔可夫决策过程的一个解决实例，并获得了对话系统中一般化策略。使用了全面的、基本 RMDP 解的两个其他方法是基于价值函数一般化 [Mausam and Weld, 2003]，和策略一般化 [Cocora et al, 2006]。这两种方法首先使用一般的马尔可夫决策过程求解器，同时使用关系决策树算法以相对简单的逻辑语言来概括解。Cdela Rosa et al (2008) 也使用 ROLLER 算法中的关系决策树从启发式规划生成的实例中学习广义策略。基于关系信封的规划（The Relational Envelope-Based Planning, REBP）[Gardioli and Kaelbling, 2003] 使用了状态空间的有限部分作为表示（即信封）。基于关系信封的规划将在信封之外通过取样逐步扩展。计算策略的目标是，通过首先产生好的起始计划并使用信封来逐步提高计划的鲁棒性。这个方法的最近的扩展是：通过使用数目不定的预测值来表示信封，而表示的复杂度能够在学习的过程中逐渐增加 [Gardioli and Kaelbling, 2008]。

267

在表 8.1 中，我们概括了本章讨论的方法。



表 8.1 我们讨论过的主要的基于模型的方法。缩写: BW= 积木问题; Cnj= 逻辑原子的联合; Q = Q 学习; PS= 优先扫除; E= 确切的; A= 近似的; PP-IPC= 计划竞赛中的计划问题 (IPC); FOF= 一阶特征; PRM= 概率关系模型; FOL= 一阶逻辑

方法	表现	类型	算法	应用
IDP (van Otterlo, 2009b)	任意	E	IDP	—
IDP (van Otterlo, 2009b)	FOL	E	IDP	物流
ReBel (Kersting et al, 2004)	Cnj	E	IDP	BW, 物流
FOVI (Hölldobler and Skvortsova, 2004)	Cnj	E	IDP	BW
FODD (Wang et al, 2008a)	FO-ADD	E	IDP	BW
FODD (Wang and Khardon, 2007)	FO-ADD	E	IDP/API	BW
(Guestrin, 2003)	PRM	A	采样	魔兽争霸, SysAdmin
FOALP (Sanner and Boutilier, 2005)	FOF	A	AVI	电梯
FOAPI (Sanner and Boutilier, 2006)	FOF	A	API	PP-IPC
(Gretton and Thiébaux, 2004a)	自适应 FOF	A	FODTR/VFA	物流 BW
(Wu and Givan, 2007)	自适应 FOF	A	贝尔曼残差	PP-IPC
FOLAO* (Karabaev and Skvortsova, 2005)	自适应 FOF	A	FO-LAO*	PP-IPC
REBP (Gardioli and Kaelbling, 2003)	信封	A	规划	BW
自适应 REBP (Gardioli and Kaelbling, 2008)	信封	A	规划	BW

8.4 无模型的解决方案

这里, 我们综述无需假设潜在的关系型马尔可夫决策过程的抽象模型的有效性的技巧。许多方法使用以下典型的 Q 学习模式:

$$\begin{array}{ccccc}
 \tilde{Q}^0 & \xrightarrow{s} & \{\langle s, a, q \rangle\} & \xrightarrow{I} & \tilde{Q}^1 & \xrightarrow{s} & \{\langle s, a, q \rangle\} & \xrightarrow{I} & \tilde{Q}^2 & \xrightarrow{s} & \dots \\
 \downarrow D/I & & \parallel & & \downarrow D/I & & \parallel & & \downarrow D/I & & \\
 \tilde{\Pi}^0 & \longrightarrow & \{\langle s, a, q \rangle\} & & \tilde{\Pi}^0 & \longrightarrow & \{\langle s, a, q \rangle\} & & \tilde{\Pi}^0 & \longrightarrow & \{\langle s, a, q \rangle\}
 \end{array}$$

这里, 一个起始的抽象 Q 函数可用于从关系型马尔可夫决策过程中获得 (s) 有偏的学习样本。这些样本将用于学习 (或推导 (I)) 一个新的 Q 函数结构。策略能够从当前的 Q 函数中计算 (或推导 (D)) 出来。将在下一节讨论该方案上受限制的变化是修复逻辑抽象级别 (例如 Q) 和只在关系马尔可夫决策过程上取样, 来获得好的估计值。

8.4.1 固定泛化的价值函数学习

几种技术使用关系的形式来创造潜在的关系型马尔可夫决策过程的紧凑表示, 固定抽象级别, 并运用标准值学习算法, 例如 Q 学习。在 CARCASS 表达中为三块积木世界的样本抽象示例 [van Otterlo, 2003, 2004]:

```

state ($_1):      (on(A,B),on(B,floor),on(C,floor),A ≠ B,B ≠ C)
actions ($_{11}, \dots, \$_{13}): move(A,C),move(C,A),move(A,floor)
state ($_2):      (on(A,floor),on(B,floor),on(C,floor), A ≠ B,B ≠ C,A ≠ C)
actions ($_{21}, \dots, \$_{26}): move(A,B),move(B,A),move(A,C), move(C,A),move(B,C),move(C,B)
state ($_3):      (on(A,B),on(B,C),on(C,floor),A ≠ B,B ≠ C,C ≠ floor)
actions ($_{31}):  move(A,floor)

```

这个算法将潜在关系型马尔可夫决策过程转化为一个更加小的抽象马尔可夫决策过程



这个方法能够通过强化学习算法及相关的改进来求解。例如，抽象状态 S_3 模拟了所有的情况，包括将所有的积木压入堆栈的情况，唯一可能的动作是将顶部的块移到底部 (A_{31})。在这种情况下，三个抽象状态泛化了 13 个关系型马尔可夫决策过程状态。抽象层是状态-动作空间的确切的聚合（即划分）[Gordon, 1995]。[van Otterlo, 2004] 在 Q 学习设置中使用了这些方法，在基于模型的方式中，例如优先扫除 [Moore and Atkeson, 1993]，学习了抽象状态之间的转换模型。本质上，学习到的知识是表示法中描述过的所有的策略中的最佳抽象策略。这些策略空间能够从领域专家处获得，也可以通过其他方式获得。

两个状态相互紧密联系的方式是 [Kersting and De Raedt, 2004] 的 LOMDP 和 [Morales, 2003] 的 rQ 学习。LOMDP 的抽象层与 CARCASS 是非常相似的，它们使用 Q 学习和逻辑 TD(λ) 算法为抽象状态学习状态值。rQ 框架是基于抽象状态 (r 状态) 和抽象动作 (r 动作) 的单独定义。 r 状态和 r 动作的产品空间带来了一个新的抽象的状态-动作空间，而这个空间将覆盖整个 Q 学习的范围。所有的方法都能够用于学习领域内的最优策略，其中，先验知识是存在的。这些方法也可以用作其他学习方法的部分。例如，在给定分层策略 (hierarchical policy) 中学习子策略。[Wang et al, 2008b] 报告了马尔可夫逻辑网络 (Markov logic network) 的相关成果。

已经报告了自动生成抽象的初步调查 [Song and Chen, 2007, 2008]，[Morales, 2004] 采用了动作克隆从子优化路径中学习 r 动作，而这些动作是人类专家产生的。AMBIL [Walker et al, 2007] 也从路径中学习了抽象状态-动作对；给出一个抽象层，与 CARCASS 类似，AMBIL 估计出一个逼近模型，并产生一个新的状态-动作对。每一个值学习迭代生成一个新的表示法。

269

固定抽象的一个变体是将抽象状态用作价值函数逼近的关系特征。每一个抽象状态可以看作一个二值的特征；一个状态可以或不可以被包括。[Walker et al, 2004] 第一次产生了一个半随机的关系特征的集合，并学习了特征的线性组合的权重用来表示每一个动作的权重。一个相关技术是关系时序差分 (rTD) 学习算法，见 [Asgharbeygi et al, 2006]。与抽象状态相似，这个算法运用了一系列的概念定义，这些概念定义必须由设计者提供并将功能分配给它们中的每一个。对这些概念来说，价值函数表示为这些概念功能的线性组合，并采用概念发生的次数作为权重。表 8.2 概括了本节内容。

表 8.2 带有静态一般化的无模型的主要方法。图例：BW= 积木问题；Cnj= 逻辑原子的结合；Q=Q 学习；PS= 优先扫除；TD=TD 学习；RF= 关系特征

方法	表现	算法	应用
CARCASS (van Otterlo, 2003, 2004)	CARCASS Cnj + 否定	Q, PS	BW, 井字游戏
LOMDP (Kersting and De Raedt, 2004)	Cnj	Q, TD	BW
RQ (Morales, 2003) (Walker et al, 2004)	Cnj + 否定 RF	Q	Taxi, 棋；网格 中型机器人联赛
rTD (Asgharbeygi et al, 2006)	RF	TD	井字游戏，迷你国际象棋

8.4.2 带自适应泛化的价值函数

对于自适应泛化来说，在学习期间改变表示法，例如 PIAGET-3 算法，ILP 方法通常用



于状态-动作样本,而这些样本来自于公式(8.4)中的关系型马尔可夫决策过程步骤。在这一节,我们将介绍三种方法:1)用于构建逻辑抽象的方法,例如价值函数;2)基于距离和核的方法;3)基于概率技巧的方法。

1. 学习价值函数抽象

如前所述, [Džeroski et al, 1998] 开发的 Q-RRL 方法是关系型马尔可夫决策过程中无模型强化学习的第一个方法。Q 学习和 ILP 算法的直接合并可以用于 Q 函数的泛化,并在小的确定的积木世界上测试过。Q-RRL 方法收集了以状态-动作对形式存在的并拥有对应 Q 值的经验。在一个阶段中,根据当前的策略采取动作,并基于当前的 Q 树。在每一个阶段之后,一个决策树将从样本集合中被推导出来,见图 8.3。Q 树能够使用背景知识,例如塔的高度和数量、积木的数量等。

然而, Q 函数的一个问题是,这些方法隐式地对到目标的距离进行了编码,它们依赖于关系型马尔可夫决策过程系列方法中的领域规模。一个 Q 函数代表了多于选择一个最优动作的信息。P 学习能够用于从当前的 Q 函数和一个训练集中学习策略 [Džeroski et al, 2001]。对于每一个状态 s 来说,其发生在训练集合中,那个状态中所有可能的动作都被评估了:如果 $a = \operatorname{argmax}_a Q(s, a')$, 那么 P 值将计算为 $P(s, a) = 1$; 如果条件不满足,那么 P 值将计算为 $P(s, a) = 0$ 。P 树表示了与 Q 树相关的最好的策略。总的来说,这些算法不是非常复杂,并且对于不同数目的块来说在所有的领域上泛化。[Lecoeuche, 2001] 独立地显示了相似的结果。[Cole et al, 2003] 采用了与 Q-RRL 相似的设置。但是, [Lloyd, 2003] 将当前的表示法语言升级为高阶逻辑 (Higher-Order Logic, HOL)。

[Driessens et al, 2001] 开发了 Q-RRL 的升级扩展版本: TG- 算法。这个算法能够视为 G 算法的关系扩展版本 [Chapman and Kaelbling, 1991], 并且这个算法在逐步地构建 Q 树。Q 树的每一片叶子通过与 Q 值相关的统计学指标和正样本的数量而增加了。一个节点在看到足够的例子时就会分裂,对节点统计数据的测试变得非常显著。此机制消除了存储 (storing)、检索 (retrieving) 和更新 (updating) 单个样本的需要。生成新的测试要比在建议的情况下复杂得多,因为基本上可能分裂的数量是无限的,并且随着前面节点中引入的变量个数的增加,树也随之大幅度增长。

在接下来的升级 TG 方法的 TGR 算法的过程中, [Ramon et al, 2007] 通过增加一棵树的重构操作来解决不可逆的问题。这些方法包括页修剪 (leaf pruning)、子树修剪 (subtree pruning) 和内部点的修改。为了执行这些操作,统计数据现在存储在树的所有节点中。在构建和重构树的过程中,要特别注意树中的变量。[Dabney and McGovern, 2007] 开发了另外一个方法:从一开始就使用重构操作,即关系型 U 树算法。[McCallum, 1995] 开发了 U 树的关系扩展版本: rU 树算法。因为 rU 树算法 (rUTree) 是基于样例的,正如 TGR 算法中一样,测试可以在需要分裂时再生,这样就不必为所有节点保存统计信息。与 [Walker, et al, 2004] 的方法相似, rU 树算法的另一个有趣的方式是使用随机取样,在分裂节点时处理大量可能的测试。结合最后两个方面,展示了一个有趣的 TG 算法与 TGR 算法之间的区别。然而, TG 算法必须保持所有的统计和考虑所有的测试,而 rU 树算法只考虑有限的被采用过的可能的测试集合。相反, rU 树算法必须经常重新计算统计值。

最后, [Mellor, 2008] 开发的一阶 XCS (即 FOXCS) 是一个基于学习能力的分类系统 (例如 [Lanzi, 2002]), 而这个分类系统具有关系型规则 (Lanzi, 2002)。FOXCS 算法的策略表示类似于 CARCASS 算法,但每个规则的准确性都会增加且每一次需要一个动作,包括

对当前的状态-动作对的所有的规则的考虑。通过一个进化学习算法 (evolutionary learning algorithm), 精度可以用于动作选择和规则的自适应 (和创建)。积木世界实例的实验结果表明, FOXCS 算法可以与其他方法 (例如 TG 算法) 进行竞争。

2. 通过距离与核进行一般化

除了构建逻辑抽象之外, 一些方法还使用其他方法来一般化和建模成为关系解释的状态。

[Driessens and Ramon, 2003] 开发的基于关系实例的回归方法使用了 [Aha, 1991] 在基础的关系状态上开发的基于实例的学习。Q 函数是被一组精心选择的有经验的样本所代表的。为了查找新遇到的状态-动作对的值, 将计算一个当前对与已经存储的对之间的距离, 新对的 Q 值是这样被计算出来的: 所代表的对的 Q 值的平均值。需要特别注意维护正确的样本集, 比如扔掉、更新和添加新的样本。在关系域的挑战之前, Q 学习的基于样本的回归被用于建议表征, 并定义成两个解释之间的合适距离。对于基于关系实例的回归方法来说, 一个特定领域的距离在事先被定义了。例如, 在积木问题中, 通过比较状态中的块和最后的编辑距离 (例如, 从一个状态到另一个状态需要多少动作), 两个状态之间的距离通过第一个重命名的变量计算出来。因为表示仅仅由基本状态和动作组成, 不使用其他背景知识或声明性偏置。[García-Durán et al, 2008] 使用了一个在基于策略的算法中的更加一般化的基于样本的方法。在机器人的操作任务中, [Katz et al, 2008] 使用一个基于关系实例的算法。其相似性度量是这样定义的: 由关系表示诱导的同构子图来表示。

后来, [Driessens and Dzeroski, 2005] 将 TG 算法和基于关系实例的回归方法合并了, 利用两种方法的长处。TG 算法建造了一个价值函数外在的结构化模型。实际上, 仅仅能建立一个粗略的逼近值。基于关系实例的回归方法不依赖于语言偏见, 基于实例的性质更适合于回归。但是, 这个算法确实因为需要大量必须样本而带来了负面的影响。合并的算法 TRENDI 建立树状的结构, 这与 TG 算法相似, 而采用一种在树的叶子的基于实例的表示法。正因为如此, 需要新的分裂标准, TG 算法和基于关系实例的回归方法的 (语言) 偏置对于 TRENDI 算法来说是必要的。然而, 在确定性的积木问题的例子中, 新算法在某些方面 (例如计算时间) 的表现比其父级的技术更好。请注意, rU 树算法算法是基于实例的表示法的集合, 该算法以树的形式集合了逻辑抽象 (logical abstraction) 层。尚没有比较被报道。[Rodrigues et al, 2008] 最近对基于关系实例的回归方法的在线动作进行了研究。结果表明, 当每个样本更新时, 系统中样本的数量减少了。

不同于基于关系实例的回归方法, [Gärtner et al, 2003] 在 KBR 算法上采取更加原则性的做法: 使用关系状态之间的距离, 且在关系强化学习中对价值函数逼近使用基于图的核和高斯过程 (Gaussian process)。将每一个动作-状态对表示为一个图, 产品核能够被这类的图所定义。内核被包装成一个高斯径向基函数 (Gaussian radial basis function), 内核可以被调整到调节一般化的数量。

3. 概率的方法

另外两个结构自适应的算法学习和使用关于环境的概率信息来优化动作。但是, 目的是不相同的。[Sanner, 2005] 发现: SVRRL 算法的目标不打折, 有限的区域内有一个单一的终端奖励。这样就可以将价值函数视为成功的概率, 这样就可以表示为关系朴素贝叶斯网络。这个网络的结构和参数是可以同时学习的。这些参数可以用基于最大似然法 (maximum likelihood) 的标准技术来计算。如果这些算法比独立特征的估计更有用, SVRRL 的两种结

构学习方法和关系特征（基础关系原子）可以合并成为联合特征。[Sanner, 2006] 发现，一个 DM-SVRRL 算法的扩展版本使用数据挖掘的方法来聚焦结构学习。在结构学习中，只有部分状态空间被经常访问。结构学习的这个特点可以用于发现经常共同出现的特征，并转化为联合特征。结构学习的这个特点可以用于建造更加大的特征。

SVRRL 不是基于 TD 学习算法，而是基于概率推理（probabilistic reasoning）。MARLIE [Croonenborghs et al, 2007b] 也使用了概率技巧，但是将这些技巧用于转换模型学习。紧接着 CARCASS 算法，它是非常少的关系型的基于模型的强化学习算法之一。这是算法能够学习基础关系原子如何从一种状态转变到另一种状态。对于每一棵概率树来说，增量学习是通过改进的 TG 算法来实现的。给予当前的状态和动作，这样的树代表如下含义：对于每一个基础实例（ground instance）来说，整个实例在下一个状态中的概率是真的。使用这个模型，我们将使用现有的稀疏采样（sparse sampling）技术来展望未来的一些步骤。原始的 TG 算法用于存储 Q 值函数。

表 8.3 总结了一些主要的特征。方法之间的详细（实验性）的比较仍然是一个待完成的艰巨的任务。方法的一个关键方面是表示和动作学习的结合。固定的抽象级别可以提供收敛性的保证，但不灵活。增量算法（例如 TG 算法）和重构手段（例如 uTree 算法和 TGR 算法）以增加计算复杂度和广泛的数据记录为代价，提供了越来越灵活的函数逼近。

所有方法中的一个重要方面是用于样本的表示，以及如何将样本一般化为抽象。基于逻辑抽象的方法具有许多显著的优点：1）它们更容易通过变量的使用来概括不同域大小的问题。2）抽象的通常是更容易理解的和可迁移的。另一个方面来说，逻辑抽象不太适合细粒度回归。对于一些简单的积木问题来说，TG 这样的算法具有严重的困难。高度关系化的问题（例如积木问题）需要在价值函数中设置复杂的模型，而在典型的强化学习过程中学习这些模式是困难的。其他方法基于其基于例子的表示法的估计值，因为这些方法可以提供更平滑的价值函数的近似值、核或一阶特征是更加适合的。

273

表 8.3 主要的带自适应一般化的无模型方法。缩写：BW= 积木问题；IB= 基于实例的；NB= 朴素贝叶斯；HOL= 高阶逻辑

方法	表现	算法	应用
Q-RRL (Džeroski et al, 1998)	树	Q 学习	BW
(Lecoeuche, 2001)	决策列表	Q 学习	对话
(Cole et al, 2003)	HOL	Q 学习	BW
TG (Driessens et al, 2001)	增量树	Q 学习	BW
TGR (Ramon et al, 2007)	自适应树	Q 学习	BW
rUTree (Dabney and McGovern, 2007)	自适应树	Q 学习	BW, sume-Go
RIB (Driessens and Ramon, 2003)	IB	Q 学习	BW
TRENDI (Driessens and Džeroski, 2005)	树 + IB	Q 学习	BW
KBR (Gärtner et al, 2003)	核	Q 学习	BW
MARLIE (Croonenborghs et al, 2007b)	可观察规则	基于模型	BW
SVRRL (Sanner, 2005)	关系型 NB	贝叶斯	西洋双陆棋

8.4.3 基于策略的求解技巧

基于策略的方法具有简单的结构：这些方法开始于策略结构 \tilde{I}^0 ，通常与基础 RMDP 的交互生成样本 (S)，生成 (D) 一个新的抽象策略 \tilde{I}^1 ，以此类推。通常的结构如下：

$$\tilde{I}^0 \xrightarrow{s} \{\langle s, a, q \rangle\} \xrightarrow{I} \tilde{I}^1 \xrightarrow{s} \{\langle s, a, q \rangle\} \xrightarrow{I} \tilde{I}^2 \longrightarrow \dots$$

所使用的表示法通常是我们以前提出的类似决策-规则的结构。价值学习的一个重要的区别是： \tilde{Q} 的外在表示法是不需要的。这些近似策略迭代算法（approximate policy iteration algorithm）的每次循环中，目前的策略是用来收集有用的学习经验，这可以是状态动作对的样本，或由该策略收集的奖励数，然后用来产生一个新策略的结构。

1. 进化关系策略搜寻

基于策略的方法的第一种类型是进化的方法，而这些方法已经用于建议强化学习（propositional RL）[Moriarty et al, 1999]。这些方法的显著特点是通常保持一个群体的（即一个集合的）策略结构，基于运用于这些问题的操作，这些方法为每一个策略或者策略规则分配一个单值的配合（single-valued fitness）。这些配合用于合并和修改策略，从而直接从策略空间中搜索。

274

Grey 系统 [Muller and van Otterlo, 2005] 使用简单的关系决策表策略来发展积木问题的策略。GAPI [van Otterlo and De Vuyst, 2009] 是类似于遗传算法的方法，并演化出概率关系策略（probabilistic relational policy）。[Guestrin, 2003] 在实时策略 FreeCraft 领域中采用了与遗传优化（Genetic Programming, GP）相关的方法 [Guestrin et al, 2003a]。结果表明，虽然在状态数目稀少的时候会发生困难，以上方法与 [Guestrin et al, 2003a] 的方法不相上下。[Castilho et al, 2004] 聚焦于 STRIPS 规划问题。但是，与 Grey 等方法不相同的是，每个染色体（chromosome）编码一个完整的规划，这意味着该方法在规划空间可以搜索。[Kochenderfer, 2003 ; Levine and Humphreys, 2003] 在策略空间中搜索，并同时使用 GP 算法。Levine 和 Humphreys 从规划器产生的最优规划中学习决策列表策略，并用于策略受限的规划器（planner）。通过同时产生可以相互调用的子策略，Kochenderfer 在策略中允许层级结构（hierarchical structure）。最后，[Baum, 1999] 描述了 Hayek 机器，而 Hayek 机器能够使用进化的方法来为积木问题学习策略。一个附加方法（FOXCS 算法）已经在上一节中介绍了。

2. 策略搜索的分类

第二种方法使用 ILP 算法从采样的状态-动作对中学习策略的结构。这实际上是将强化学习的步骤转化成一系列的监督学习（分类）的任务。通过使用以前的策略结构，一个抽象策略将不断重复地从状态-动作对中取样。具有挑战性的任务是：如何能够获得好的样本？即可以通过从最优路径采样的方法（例如，人类专家、规划算法），或者通过从当前策略进行聪明轨迹采样。两种方法都是 PIAGET-3：将结构和参数学习放置在同一个算法中。

[Yoon et al, 2002] 采用基于模型的方法从规划器产生的最优路径（optimal trace）中推导策略。这些算法可以看作当前工作的一个扩展，扩展到随机领域（stochastic domain）中，详见 [Martin and Geffner, 2000 ; Khardon, 1999]。[Khardon, 1999] 研究了在非折扣的和基于目标的规划领域推导确定性的策略，并从包含一定数量的策略的样本中证明了一般的 PAC 上界和下界。

[Fern et al, 2006] 统一和扩展了 LRW-API 方法，将刚刚提到的方法提升为一个可以实际操作的软件。与 [Martin and Geffner, 2000] 的方法相似，LRW-API 是基于概念语言（分类语法）的。与 [Yoon et al, 2002] 的方法相似，LRW-API 是基于复杂的、概率的规划域的 LWR-API 方法通过共享主要的思想，推导出了当前策略，即近似策略迭代（API），进而

为了推导一个更好的策略而偏置样本的采集。LRW-API 算法的与早期方法相关的两个主要的改进之处包括：样本的取样过程和拔靴法过程（bootstrapping process）。关于第一个方法，
[275] LRW-API 采用了策略部署对当前的策略进行取样 [Boyan and Moore, 1995]。也就是说，通过取样长度 h 的轨迹 w 的方式，这些方法为一个状态的当前策略估计所有的动作值。其中，每一个轨迹是状态的结果，并为 $h-1$ 的更多步骤产生策略。注意，这需要一个可以从任何时间和任何状态进行采样的模拟器。LRW-API 算法的第二个主要提升是拔靴法过程，这意味着从随机事件中学习（LRW）。这个想法是先从简单问题开始，然后通过迭代的方式来解决越来越复杂的问题。通过基本的关系马尔可夫决策过程，一个长度为 n 的随机游走（random walk）算法将产生一个问题实例。当扩展到 n 个问题的时候，问题变得更加复杂了。

3. 策略梯度方法

最后一种基于策略的技术是基于策略梯度的方法 [Sutton et al, 2000]。为了这个目标，我们需要一个参数化的关系策略并使用策略参数的梯度来优化策略。

[Itoh and Nakamura, 2004] 描述了部分可观察的关系马尔可夫决策过程的第一个方法。其中，策略用关系决策列表来表示。该手动编码的策略利用了一个字节数量有限内存。该算法在迷宫类的域中进行了测试：规划有时候是有作用的，问题是学习何时算法是有用的。这是通过随机策略来实现的。其中，策略规则（policy rule）的概率用于探索和通过梯度下降技术（gradient descent techniques）来进行算法的学习。[Gretton, 2007a] 开发了 ROPG 算法，而 ROPG 算法可以学习非马尔可夫奖励域的时间扩展策略。策略梯度可以自然使用，生成策略规则的方法有两种：基于取样的方法和从问题特征计算出来的规则。总的来说尽管收敛很慢，在电梯分配问题中，RPOG 算法学会了有用的通用策略。近来，在非参数的策略梯度（Non-Parametric Policy Gradient, NPPG）方法中，[Kersting and Driessens, 2008] 采用了梯度推进（gradient boosting）的想法。NPPG 方法建立价值函数的回归模型。因此，扩展了学习中的表示法（PIAGET-3）。因为这个方法只是提高了计算梯度的水平，以类似的方式，这个方法为建议域（propositional domain）和关系域（relational domain）工作。

8.5 模型、层级、偏置

本章前两部分对传统的基于模型和无模型的算法关系型的算法的最新进展进行了研究。在本节，我们将探讨一下关系型马尔可夫决策过程框架的其他方法的关系型技巧。三组：关系型马尔可夫决策过程的概率模型、用于策略的结构、学习器的偏置。
[276]

表 8.4 主要的无模型的基于策略的方法。缩写：BW= 积木问题；PG= 策略梯度；Q= Q 学习；PS= 优先扫除；TD=TD 学习；DL= 决策队列；GD= 梯度下降

方法	表现	算法	应用
Grey (Muller and van Otterlo, 2005)	DL	演化	BW
FOX-CS (Mellor, 2008)	规则	Q 演化	BW
(Gearhart, 2003)	PRM	GP	魔兽争霸
(Yoon et al, 2002)	DL	规划	BW
LRW-API Fern et al (2007)	DL	快速走子	规划
(Itoh and Nakamura, 2004)	可观察规则	PG/GD	迷宫
RPOG (Gretton, 2007a)	规则	PG	电梯规划
NPPG (Kersting and Driessens, 2008)	递归梯	PG	BW
GAPI (van Otterlo and De Vuyst, 2009)	可观察 DL	演化	BW, Goldfinder

1. 学习世界的模型

学习世界模型是学习器能做的最有用的事情之一。转换模型体现了有关环境的知识，可以以各种方式加以利用。因为基于模型的动态规划算法，这些模型可用于更有效的强化学习的算法。进一步来说，这些模型可以转换到相似环境下的其他模型。有几种方法可以从交互中学习一般的运算符模型。相反，在 8.3 节中，所有基于模型的方法将确保获得一个完整的逻辑性的模型。

通常来说，模型学习意味着学习一般的算子描述，例如前面已经讨论过的 STRIPS 规则。然而，简单的模型已经可以非常有用的，我们已经看到例子，例如，MARLIE 的部分模型和 CARCASS 算法的抽象模型。另一个例子是更加特定动作的模型学习。[Morales, 2004] 使用了这个模型，并通过动作克隆学习了 r 动作。[Halbritter and Geibel, 2007] 开发了新的方法：这个方法是基于图核心 (graph kernel) 的。图核心可用于存储转换模型，而不使用逻辑抽象，例如大多数动作的形式化过程中使用的逻辑抽象。[Mourão et al, 2008] 提出的机器人相关的方法是基于核感知机 (kernel perceptron) 的，但受制于学习 (确定性的) 类似 STRIPS 方法的影响。

例如，仅仅限于学习动作的影响，计划数据中的 STRIPS 算子的学习部分是一个老的问题，几个老的工作提供部分的解决方案。例如，学习动作的影响 [Vere, 1977]。然而，随着概率结果 (probabilistic outcome) 的难度增加，学习完全模型是复杂的，因为完全模型需要从数据中学习逻辑结构和参数。记住我们先前定义的概率 STRIPS 动作。这些动作拥有两个不同的概率的结果。从数据中学习这样的描述涉及许多方面的问题。第一，预先条件和后置条件的逻辑描述必须从数据中学习。第二，学习算法必须推断出动作有多少结果。第三，必须对动作的每一结果估计概率。相近的样例是，早期的方法 [Gil, 1994; Wang, 1995] 通过与模拟的世界交互的方式，从数据中学习确定性的算子。最近出现了更多的工作，聚焦于不完全状态信息 (incomplete state information) 的内容 [Wu et al, 2005; Zhuo et al, 2007]，或者考虑样本复杂性的方面 [Walsh and Littman, 2008]。

[277]

对于一般性的关系型马尔可夫决策过程，我们需要概率模型。在建议设置 (propositional setting) 中，[Oates and Cohen, 1996] 描述了早期的学习方法。通过三步贪婪的搜寻方法，[Pasula et al, 2004] 报道了在关系型马尔可夫决策过程上下文中关系模型的第一种方法。首先，通过使用标准的 ILP 运算符的规则集合进行搜索。其次，对于一个上下文和动作来说，发现了输出结果的最佳集合。第三，这个方法通过一系列的输出学习了一个概率分布 (probability distribution)。因为这个方法从域中获取的状态 - 动作 - 状态对的数据集，学习过程是受监督的。结果是，规则对这个集合是唯一有效的，必须注意的是该领域的表示法。在积木问题和逻辑领域进行的实验显示了这种方法的鲁棒性。[Zettlemoyer et al, 2005] 通过增加噪声输出的方式扩展了这个方法，即，很难准确建模的但是一定会发生的结果 (撞到一个积木的塔或者把所有的积木都散射开来)。[Safaei and Ghassem-Sani, 2007] 介绍了另外一个方法，渐进式地结合了规划和学习。

2. 层级和学习器

层级的方法 (hierarchical approach) 可以自然地合并到关系型强化学习中，尽管出现了一些认知的结构 (和聪明的学习器 [van Otterlo et al, 2007])，到目前为止还没有多少技术被报道过。通过逻辑变量，关系型层次强化学习 (HRL) 的一个优点是子策略和目标自然地参数化。例如，在 (X, Y) 的一个积木任务，其中 X 和 Y 可以使用两个积木来实现，可以分

解成两个子任务。首先,所有的积木都必须从 X 和 Y 中移除,然后 X 和 Y 应该被移到另一个面的上面。注意,第一个任务也包含两个子任务,支持进一步分解。现在,首先学习这些子任务的策略,每个子任务的单个学习问题都要简单得多,结果技能能够被重用。进而言之,在 8.4 节,学习这些子任务可以由任何无模型的算法来完成。取决于所使用的表示法,可以将策略构造成层级结构,在更复杂的问题上促进学习。

[278]

简单的选择形式很容易对关系进行建模 [Croonenborghs et al, 2007a]。[Driessens and Blockeel, 2001] 提出一种基于 Q-RRL 算法的方法来同时学习两个目标,这可以看作一种简单的层级分解 (hierarchical decomposition) 形式。[Aycenina, 2002] 使用了原始的 Q-RRL 系统来构建一个完整的系统。给出了许多子目标,并学习了不同的策略以实现这些子目标。当学习一个更复杂的任务时,实例化的子策略可以作为新的动作。[Roncagliolo and Tadepalli, 2004] 开发了一个相关的系统,在一组示例中使用批处理来学习给定关系层级结构 (relational hierarchy) 的值。[Andersen, 2005] 使用与关系表示法相适应的 MAXQ 层级结构进行最彻底的调查。基于手动创建的子目标的层级结构,该工作使用 Q-RRL 框架来推导出本地的 Q 树和 P 树。层级关系系统 (hierarchical relational system) 的一个缺点是,它们假定层次结构是预先给定的。两个相关的系统结合规划和学习,可以认为是通过规划生成层级抽象 (hierarchical abstraction), 并使用强化学习算法来学习具体的动作策略。[Ryan, 2002] 发明的方法使用一个规划器来构建一个高级任务层级结构,强化学习是用来学习子策略的。[Grounds and Kudenko, 2005] 发明的方法也是基于相似的想法。在认知架构中,除了模型学习的以外,包括 Icarus [Shapiro and Langley, 2002] 和 SOAR 的强化学习 [Nason and Laird, 2004], 也没有多少值得报告的工作。

层级学习 (hierarchical learning) 可以看作首先分解一个策略,然后是学习。[Wooldrige, 2002] 发现,多学习器分解也是可能的。[Letia and Precup, 2001] 报告了多学习器,多学习器被建模为相互独立的强化学习器,这些学习器之间无法通信,但在相同的环境下动作。程序将指定初始的计划和关于环境的知识,复杂的动作将会推导出半马尔可夫决策过程模型,而学习是由基于选项的无模型的强化学习方法来执行的。[Hernandez et al, 2004] 开发出了相似的方法,这些方式是基于信任期望-注意模型 (belief desires-intentions model) 的。[Finzi and Lukasiewicz, 2004a] 开发了 GTGolog 语言,一种博弈论的语言 (game-theoretic language), 在马尔可夫游戏 (Markov Game) 中,这种语言将显式的学习器优化和基于博弈论的多学习器计划结合在一起。在这个方向上, [Finzi and Lukasiewicz, 2004] 引入关系型马尔可夫游戏 (relational Markov Game), 这个方法可以用来抽象多学习器的关系型马尔可夫决策过程 (RMDP) 算法,并计算纳什策略对。

3. 偏置

关系型马尔可夫决策过程的求解算法可以在很多方面得到帮助。一个基本的区别是帮助解决当前问题的方法 (包括偏置、引导等) 还是帮助解决相关的问题的方法 (迁移)。

关于第一个想法是为学习器提供一个策略,通过产生可以作为学习器经验的半最佳的动作轨迹 (semi-optimal traces of behavior), 这个策略可以引导学习器走向状态空间中有用的领域 [Driessens and Džeroski, 2002]。与指导相关的是基于人类产生的痕迹 (human generated traces) 的动作克隆的使用 [Morales, 2004; Cocora et al, 2006] [Yoon et al, 2002] 使用的最优策略、[Fern et al, 2006] 提出的随机游走方法,这个方法使用了对域实例化 (domain instantiation) 的引导探索 (guided exploration)。在后一种情况下,首先只生成简单

的实例化，并且根据当前策略的质量增加问题的难度。[Guestrin et al, 2003a] 使用了领域取样 (domain sampling) 的方法。另一种帮助学习器的方法是通过构造域本身，例如通过建立一个强大的拓扑结构的方法 [Lane and Wilson, 2005]。由于许多这些指导技术本质上是表示法独立的，所以许多现有的算法（或者带建议的算法）可以用于关系型马尔可夫决策过程 (RMDP) 之中。

第二种帮助学习器的方法是从其他任务中迁移知识。简而言之，迁移学习是将学习的知识运用到一项源任务上，用来改进相关目标的学习，但有不同的目标任务。迁移学习特别适合于持续时间、在任务和环境之间灵活变化的学习器。与其从头开始学习每一项任务，我们的目标是利用过去的经验加速学习 [Stone, 2007]。迁移是非常依赖表示法的，在关系型强化学习中使用（声明性的）FOL 形式化提供了很好的机会。 [279]

在更严格的（关系的）强化学习的设置中，有几种可能性。例如，源和目标问题在必须达到的目标中可能有所不同，但可能转换模型和奖励模型可以迁移。有时，特定的状态抽象可以在问题之间迁移 [Walsh et al, 2006]。或者，可能有关操作的一些知识可以迁移，尽管这些知识在源任务和目标任务中可能有稍微不同的效果。有时候可以迁移完整的策略。例如，积木问题中的堆栈策略可以在不同大小的世界之间迁移。另一种可能性是将解决方案迁移到子问题，例如子策略。通常在关系域 (relational domain) 中，仅由关系表示的内在性质 (intrinsic nature) 就可能发生迁移。也就是说，在许多情况下，积木问题中的 n 块任务学习的策略将工作于 $m > n$ 的情况。

到目前为止，一些方法已经接近于关系型的强化学习中更一般的迁移概念。几种方法是基于分层分解 (hierarchical decomposition) 学习技能的迁移 [Croonenborghs et al, 2007a; Torrey et al, 2007]，并使用马尔可夫逻辑网络 (Markov logic network) 来扩展后者 [Torrey et al, 2008]。迁移学习的明确调查是基于我们讨论过的方法，例如，rTD 方法 [Stracuzzi and Asgharbeygi, 2006]，或者 [Garca-Duran et al, 2008] 的研究工作——确定性规划域 (deterministic planning domain) 中基于实例的策略迁移。

8.6 现在的发展

到目前为止，我们已经讨论了一个重要的选择方法 [van Otterlo, 2009b]。在最后一部分中，我们简要地讨论有关关系型马尔可夫决策过程算法新出现的求解技术、新的一些方法和发展趋势。这些题目还标记了至少两个具有许多开放性问题 and 潜在性的领域；一般逻辑概率引擎 (logical-probabilistic engine) 和部分可观察问题。

1. 概率推理框架

我们已经讨论了许多技术的解决方案，通过升级 MPD 的表示法、求解算法和方法的关系域，能够解决关系型马尔可夫决策过程 (RMDP) 算法。另一条路线已经引起了研究者相当大的兴趣：决策理论问题可以通过一般的概率和决策的理论推理的方法来求解。这个方向从第一次观察规划开始，作为概率推理 (probabilistic inference) 的一个具体应用实例 [Toussaint, 2009]。 [280]

对于任何一个的马尔可夫决策过程，可以找到一个公式，其基本任务是发现策略 π ，并使达到目标的概率最大化。规划和概率推理之间的这种联系开辟了许多可能性，将概率规划问题 (probabilistic planning problem) 作为一般的概率推理问题 (probabilistic inference problem)，并利用优化策略寻找最优规划。

[Lang and Toussaint, 2009] 在基于 [Pasula et al, 2004] 开发的学习规则的上下文中直接实现这些思想。然后将关系概率转换模型转化为贝叶斯网络, 并采用标准推理技术进行(重新)计划。后来, 又通过融合向前延伸和向后延伸规划进行了扩展 [Lang and Toussaint, 2010]。通过转换为建议贝叶斯网络, 这些方法很容易连接到较低级别的概率推理模式, 从而形成一个集成的框架, 包括应用于真正的机械手臂的高层次的关系学习和规划与低级别的动作 [Toussaint et al, 2010]。然而, 对贝叶斯网络进行翻译的一个缺点是: 虽然这些网络的规模通常是爆炸性的, 但是规划仍然实质上是在建议层次上进行的。一个更加合理的导向是由 [Thon et al, 2009] 描述的 SRL 方法来学习概率的动作模型, 而这个模型可以直接翻译成标准的关系计划语言 (standard relational planning language)。

目前, 正在开发的一种更通用的解决方案: 将 SRL 系统扩展到具有决策理论能力的全概率编程语言中。早期有许多想法, 主要是在独立选择逻辑 (independent choice logic) [Poole, 1997], 最近出现的方法采用最先进的推理和(或)学习的技术。DT-ProbLog 语言是第一个概率决策理论规划语言, 能够有效地计算最优(近似)的规划 [Van den Broeck et al, 2010]。这些方法是基于 ProbLog 语言进行概率推理的。但是, 用奖励事实眼延伸。基于一组决策事实, DT-ProbLog 语言将计算可能的规划, 并紧凑地存储代数决策图 (Algebraic Decision Diagrams, ADD) 中的值。ADD 的优化给出了最优(近似)的决策集。

到目前为止, 该语言有效地处理了同时的动作 (simultaneous action), 即必须在一个问题中做出多个决定, 但尚未考虑到顺序的方面。例如, 可以模拟和求解这些问题, 但有效的解决方案是一个必须采取的步骤。这些方法成功地成为一个庞大的社交网络 (huge social network) 做出了直接的营销决策 (direct marketing decision)。基于马尔可夫逻辑网络 [Nath and Domingos, 2009] 的相关方法可以模拟类似的问题, 但只支持近似推理。另一种方法是基于随机逻辑规划 (stochastic logic programming) [Chen and Muggleton, 2010], 但缺乏有效的实现。另一种沿着同一方向发展的方法是基于答案集编程 (answer set programming) [Saad, 2008] 的。在所有这些新方法的核心假设是: 在 SRL 系统中, 目前的进展在解决决策问题的过程中将是可用的, 例如关系型马尔可夫决策过程方法。一个相关的方向是在编程语言中使用强化学习, 有望在这方面变得重要起来 [Simpkins et al, 2008]。

2. 关系型 POMDP

281

到目前为止, 几乎所有的方法都承担了完全可观测的一阶马尔可夫决策过程。除了扩展到更复杂的模型之外, 马尔可夫假设 (Markov assumption) 几乎至今未开发, 在 [van Otterlo, 2009b] 中我们提到一些走这条路的方法。

[Wingate et al, 2007] 提出了关系型知识表达在预测表示状态 [Littman et al, 2001] 中的第一步。虽然表示法实质上是建议的, 这些方法捕捉到的一些积木问题的结构框架比马尔可夫决策过程方法更丰富。[Zhao and Doshi, 2007] 介绍一种基于半马尔可夫扩展 (semi-Markov extension) 的方法, 用于 Haley 系统中面向网页的服务的象征性动态规划算法。尽管没有算法可以求解一阶马尔可夫决策过程问题, 这个方法可以清楚地显示捕捉有用的时间结构的形式。[Gretton, 2007a, b] 使用策略梯度方法计算具有非马尔可夫奖励的域的时间扩展策略 (temporally extended policy), 我们已经在前一章中讨论过具体的细节。

[Wang, 2007] 开发了求解一阶部分观察的马尔可夫决策过程的第一个算法。虽然使用 FOL 形式化来建立部分观察的马尔可夫决策过程模型的方面已经做了很多工作 [Geffner and Bonet, 1998; Wang and Schmolze, 2005], 其是第一个升级了现有的部分观察的马尔

可夫决策过程至一阶的求解算法。这个方法采用了 FODD 的形式，我们在前面的章节已经讨论过具体细节，并将这种方法延伸到为基于状态的观察建立模型。基于内涵动态规划算法的基于回归的备份和信念状态之间的备份，Wang 将增量剪枝算法（incremental pruning algorithm）[Kaelbling et al, 1998] 升级为一阶的形式。

最近，提出了一些额外的关于关系型部分观察的马尔可夫决策过程算法的研究工作。例如，[Lison, 2010] 的对话系统、[Wang and Khardon, 2010 ; Sanner and Kersting, 2010] 描述的内涵动态规划的部分可观察的马尔可夫决策过程的基本算法。

3. 新方法和应用

除了推理引擎和部分可观察的马尔可夫决策过程方法之外，其他方法在最近也出现了。在前面我们已经提到了进化技术 GAPI[van Otterlo and De Vuyst, 2009] 此处，[Neruda and Slusny, 2009] 提供关系型强化学习与进化技术的性能比较。

学习模型的新技术包括基于确定性的 SOAR 认知结构的动作模型 [Xu and Laird, 2010]，和在噪音的上下文的基于增量学习（incremental learning）的动作模型 [Rodrigues et al, 2010]。代表面向学习转换模型的新的步骤，可用于规划和一阶决策 – 理论回归方法。基于一系列的机器人的低层的传感器的读数，[Vargas-Govea and Morales, 2009] 提出相关的技巧学习一系列动作的语法。

事实上，诸如机器人之类的领域是关系型强化学习最有趣的应用领域。[Vargas and Morales, 2008; Hernandez and Morales, 2010] 在机器人领域将关系技巧应用于导航的目的。使用动作克隆，第一种方法从轨迹方面学习了目的 – 反应程序（teleo-reactive programs）。第二种方法结合了关系型强化学习和连续的动作。关系型强化学习的另一个有趣的区域是计算机视觉。[Hming and Peters, 2009] 报告了这个方向的开创性工作，用于物体识别。

[282]

最近，有很多关于关系型强化学习的论文，如 [van Otterlo, 2009b]。五个相关的博士论文出现了：模型辅助的方法 [Croonenborghs, 2009]、迁移学习 [Torrey, 2009]、强化学习的倾向物体的表示法 [Diuk, 2010]、高效的模型学习 [Walsh, 2010] 和一阶决策 – 理论回归 [Joshi, 2010]。

8.7 总结和展望

本章综述了关系型强化学习，并概述了在 [van Otterlo, 2009b] 中讨论过的主要技巧。我们已经讨论了大的领域，包括基于模型的算法和无模型的算法，还包括层次结构、模型、部分可观察的马尔可夫决策过程、知识表达、动态规划、关系型马尔可夫决策过程、马尔可夫决策过程、决策 – 理论回归、一阶决策 – 理论回归、象征性动态规划、一阶拟合线性优化技巧、一阶拟合策略迭代算法、基于关系实例的回归方法、基于抽象化和一般化的策略迭代、关系型强化学习等。

对于关系型强化学习来说，有许多未来的方向值得探讨。一方面，可以注意到的是，并不是很多技术都使用了有效的数据结构。一些基于模型的技术使用二进制的（或代数的）数据结构，因为关系型强化学习是一个计算复杂的任务，许多其他方法可能受益。更高效的推理引擎的使用已经开始与编程语言（例如 DT-Problog 语言）的发展相适应，但预计未来几年需要做更多工作，沿着这些线路将在 SRL 领域得到大的发展。在这个方向，POMDP 技术和概率模型的学习也可以从 SRL 的角度来探讨。

就应用领域而言，机器人技术、计算机视觉、网络和社交网络可能产生有趣的问题。机

机器人的操作和导航的连接很容易完成, 目前还没有使用很多关系型强化学习技术。特别是传感器数据的接地和锚定问题是一个很难解决的问题, 而且基本上没有解决。

参考文献

- Aha, D., Kibler, D., Albert, M.: Instance-based learning algorithms. *Machine Learning* 6(1), 37–66 (1991)
- Alpaydin, E.: *Introduction to Machine Learning*. The MIT Press, Cambridge (2004)
- Andersen, C.C.S.: Hierarchical relational reinforcement learning. Master's thesis, Aalborg University, Denmark (2005)
- Asgharbeygi, N., Stracuzzi, D.J., Langley, P.: Relational temporal difference learning. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 49–56 (2006)
- Aycenina, M.: Hierarchical relational reinforcement learning. In: *Stanford Doctoral Symposium* (2002) (unpublished)
- Baum, E.B.: Toward a model of intelligence as an economy of agents. *Machine Learning* 35(2), 155–185 (1999)
- Baum, E.B.: *What is Thought?* The MIT Press, Cambridge (2004)
- Bergadano, F., Gunetti, D.: *Inductive Logic Programming: From Machine Learning to Software Engineering*. The MIT Press, Cambridge (1995)
- Bertsekas, D.P., Tsitsiklis, J.: *Neuro-Dynamic Programming*. Athena Scientific, Belmont (1996)
- Boutilier, C., Poole, D.: Computing optimal policies for partially observable markov decision processes using compact representations. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 1168–1175 (1996)
- Boutilier, C., Dean, T., Hanks, S.: Decision theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research* 11, 1–94 (1999)
- Boutilier, C., Dearden, R.W., Goldszmidt, M.: Stochastic dynamic programming with factored representations. *Artificial Intelligence* 121(1-2), 49–107 (2000)
- Boutilier, C., Reiter, R., Price, B.: Symbolic dynamic programming for first-order MDP's. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 690–697 (2001)
- Boyan, J.A., Moore, A.W.: Generalization in reinforcement learning: Safely approximating the value function. In: *Proceedings of the Neural Information Processing Conference (NIPS)*, pp. 369–376 (1995)
- Brachman, R.J., Levesque, H.J.: *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers, San Francisco (2004)
- Castilho, M.A., Kunzle, L.A., Lecheta, E., Palodeto, V., Silva, F.: An Investigation on Genetic Algorithms for Generic STRIPS Planning. In: Lemaître, C., Reyes, C.A., González, J.A. (eds.) *IBERAMIA 2004. LNCS (LNAI)*, vol. 3315, pp. 185–194. Springer, Heidelberg (2004)
- Chapman, D., Kaelbling, L.P.: Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 726–731 (1991)
- Chen, J., Muggleton, S.: Decision-theoretic logic programs. In: *Proceedings of the International Conference on Inductive Logic Programming (ILP)* (2010)
- Cocora, A., Kersting, K., Plagemann, C., Burgard, W., De Raedt, L.: Learning relational navigation policies. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS* (2006)
- Cole, J., Lloyd, J.W., Ng, K.S.: Symbolic learning for adaptive agents. In: *Proceedings of the Annual Partner Conference, Smart Internet Technology Cooperative Research Centre* (2003), http://csl.anu.edu.au/jwl/crc_paper.pdf
- Croonenborghs, T.: Model-assisted approaches for relational reinforcement learning. PhD thesis, Department of Computer Science, Catholic University of Leuven, Belgium (2009)

- Croonenborghs, T., Driessens, K., Bruynooghe, M.: Learning relational options for inductive transfer in relational reinforcement learning. In: Proceedings of the International Conference on Inductive Logic Programming (ILP) (2007a)
- Croonenborghs, T., Ramon, J., Blockeel, H., Bruynooghe, M.: Online learning and exploiting relational models in reinforcement learning. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 726–731 (2007b)
- Dabney, W., McGovern, A.: Utile distinctions for relational reinforcement learning. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 738–743 (2007)
- de la Rosa, T., Jimenez, S., Borrajo, D.: Learning relational decision trees for guiding heuristic planning. In: Proceedings of the International Conference on Artificial Intelligence Planning Systems (ICAPS) (2008)
- De Raedt, L.: Logical and Relational Learning. Springer, Heidelberg (2008)
- Dietterich, T.G., Flann, N.S.: Explanation-based learning and reinforcement learning: A unified view. *Machine Learning* 28(503), 169–210 (1997)
- Diuk, C.: An object-oriented representation for efficient reinforcement learning. PhD thesis, Rutgers University, Computer Science Department (2010)
- Diuk, C., Cohen, A., Littman, M.L.: An object-oriented representation for efficient reinforcement learning. In: Proceedings of the International Conference on Machine Learning (ICML) (2008)
- Driessens, K., Blockeel, H.: Learning Digger using hierarchical reinforcement learning for concurrent goals. In: Proceedings of the European Workshop on Reinforcement Learning, EWRL (2001)
- Driessens, K., Džeroski, S.: Integrating experimentation and guidance in relational reinforcement learning. In: Proceedings of the Nineteenth International Conference on Machine Learning, pp. 115–122 (2002)
- Driessens, K., Džeroski, S.: Combining model-based and instance-based learning for first order regression. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 193–200 (2005)
- Driessens, K., Ramon, J.: Relational instance based regression for relational reinforcement learning. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 123–130 (2003)
- Driessens, K., Ramon, J., Blockeel, H.: Speeding Up Relational Reinforcement Learning Through the Use of an Incremental First Order Decision Tree Learner. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 97–108. Springer, Heidelberg (2001)
- Džeroski, S., De Raedt, L., Blockeel, H.: Relational reinforcement learning. In: Shavlik, J. (ed.) Proceedings of the International Conference on Machine Learning (ICML), pp. 136–143 (1998)
- Džeroski, S., De Raedt, L., Driessens, K.: Relational reinforcement learning. *Machine Learning* 43, 7–52 (2001)
- Feng, Z., Dearden, R.W., Meuleau, N., Washington, R.: Dynamic programming for structured continuous Markov decision problems. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pp. 154–161 (2004)
- Fern, A., Yoon, S.W., Givan, R.: Approximate policy iteration with a policy language bias: Solving relational markov decision processes. *Journal of Artificial Intelligence Research (JAIR)* 25, 75–118 (2006); special issue on the International Planning Competition 2004
- Fern, A., Yoon, S.W., Givan, R.: Reinforcement learning in relational domains: A policy-language approach. The MIT Press, Cambridge (2007)
- Fikes, R.E., Nilsson, N.J.: STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2(2) (1971)
- Finney, S., Gardiol, N.H., Kaelbling, L.P., Oates, T.: The thing that we tried Didn't work very well: Deictic representations in reinforcement learning. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pp. 154–161 (2002)
- Finzi, A., Lukasiewicz, T.: Game-theoretic agent programming in Golog. In: Proceedings of the European Conference on Artificial Intelligence (ECAI) (2004a)

- Finzi, A., Lukasiewicz, T.: Relational Markov Games. In: Alferes, J.J., Leite, J. (eds.) JELIA 2004. LNCS (LNAI), vol. 3229, pp. 320–333. Springer, Heidelberg (2004)
- García-Durán, R., Fernández, F., Borrajo, D.: Learning and transferring relational instance-based policies. In: Proceedings of the AAAI-2008 Workshop on Transfer Learning for Complex Tasks (2008)
- Gardiol, N.H., Kaelbling, L.P.: Envelope-based planning in relational MDPs. In: Proceedings of the Neural Information Processing Conference (NIPS) (2003)
- Gardiol, N.H., Kaelbling, L.P.: Adaptive envelope MDPs for relational equivalence-based planning. Tech. Rep. MIT-CSAIL-TR-2008-050, MIT CS & AI Lab, Cambridge, MA (2008)
- Gärtner, T., Driessens, K., Ramon, J.: Graph kernels and Gaussian processes for relational reinforcement learning. In: Proceedings of the International Conference on Inductive Logic Programming (ILP) (2003)
- Gearhart, C.: Genetic programming as policy search in Markov decision processes. In: Genetic Algorithms and Genetic Programming at Stanford, pp. 61–67 (2003)
- Geffner, H., Bonet, B.: High-level planning and control with incomplete information using pomdps. In: Proceedings Fall AAAI Symposium on Cognitive Robotics (1998)
- Gil, Y.: Learning by experimentation: Incremental refinement of incomplete planning domains. In: Proceedings of the International Conference on Machine Learning (ICML) (1994)
- Gordon, G.J.: Stable function approximation in dynamic programming. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 261–268 (1995)
- Gretton, C.: Gradient-based relational reinforcement-learning of temporally extended policies. In: Proceedings of the International Conference on Artificial Intelligence Planning Systems (ICAPS) (2007a)
- Gretton, C.: Gradient-based relational reinforcement learning of temporally extended policies. In: Workshop on Artificial Intelligence Planning and Learning at the International Conference on Automated Planning Systems (2007b)
- Gretton, C., Thiébaux, S.: Exploiting first-order regression in inductive policy selection. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pp. 217–225 (2004a)
- Gretton, C., Thiébaux, S.: Exploiting first-order regression in inductive policy selection (extended abstract). In: Proceedings of the Workshop on Relational Reinforcement Learning at ICML 2004 (2004b)
- Groote, J.F., Tveretina, O.: Binary decision diagrams for first-order predicate logic. *The Journal of Logic and Algebraic Programming* 57, 1–22 (2003)
- Grounds, M., Kudenko, D.: Combining Reinforcement Learning with Symbolic Planning. In: Tuyls, K., Nowe, A., Guessoum, Z., Kudenko, D. (eds.) ALAMAS 2005, ALAMAS 2006, and ALAMAS 2007. LNCS (LNAI), vol. 4865, pp. 75–86. Springer, Heidelberg (2008)
- Guestrin, C.: Planning under uncertainty in complex structured environments. PhD thesis, Computer Science Department, Stanford University (2003)
- Guestrin, C., Koller, D., Gearhart, C., Kanodia, N.: Generalizing plans to new environments in relational MDPs. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 1003–1010 (2003a)
- Guestrin, C., Koller, D., Parr, R., Venkataraman, S.: Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research (JAIR)* 19, 399–468 (2003b)
- Halbritter, F., Geibel, P.: Learning Models of Relational MDPs Using Graph Kernels. In: Gelbukh, A., Kuri Morales, Á.F. (eds.) MICAI 2007. LNCS (LNAI), vol. 4827, pp. 409–419. Springer, Heidelberg (2007)
- Hanks, S., McDermott, D.V.: Modeling a dynamic and uncertain world I: Symbolic and probabilistic reasoning about change. *Artificial Intelligence* 66(1), 1–55 (1994)
- Guerra-Hernández, A., Fallah-Seghrouchni, A.E., Soldano, H.: Learning in BDI Multi-Agent Systems. In: Dix, J., Leite, J. (eds.) CLIMA 2004. LNCS (LNAI), vol. 3259, pp. 218–233. Springer, Heidelberg (2004)

- Hernández, J., Morales, E.F.: Relational reinforcement learning with continuous actions by combining behavioral cloning and locally weighted regression. *Journal of Intelligent Systems and Applications* 2, 69–79 (2010)
- Häming, K., Peters, G.: Relational Reinforcement Learning Applied to Appearance-Based Object Recognition. In: Palmer-Brown, D., Draganova, C., Pimenidis, E., Mouratidis, H. (eds.) EANN 2009. Communications in Computer and Information Science, vol. 43, pp. 301–312. Springer, Heidelberg (2009)
- Hölldobler, S., Skvortsova, O.: A logic-based approach to dynamic programming. In: Proceedings of the AAAI Workshop on Learning and Planning in Markov Processes - Advances and Challenges (2004)
- Itoh, H., Nakamura, K.: Towards learning to learn and plan by relational reinforcement learning. In: Proceedings of the ICML Workshop on Relational Reinforcement Learning (2004)
- Joshi, S.: First-order decision diagrams for decision-theoretic planning. PhD thesis, Tufts University, Computer Science Department (2010)
- Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101, 99–134 (1998)
- Kaelbling, L.P., Oates, T., Gardiol, N.H., Finney, S.: Learning in worlds with objects. In: The AAAI Spring Symposium (2001)
- Karabaev, E., Skvortsova, O.: A heuristic search algorithm for solving first-order MDPs. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI) (2005)
- Karabaev, E., Rammé, G., Skvortsova, O.: Efficient symbolic reasoning for first-order MDPs. In: ECAI Workshop on Planning, Learning and Monitoring with Uncertainty and Dynamic Worlds (2006)
- Katz, D., Pyuro, Y., Brock, O.: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation. In: Proceedings of Robotics: Science and Systems IV (2008)
- Kersting, K., De Raedt, L.: Logical Markov decision programs and the convergence of $TD(\lambda)$. In: Proceedings of the International Conference on Inductive Logic Programming (ILP) (2004)
- Kersting, K., Driessens, K.: Non-parametric gradients: A unified treatment of propositional and relational domains. In: Proceedings of the International Conference on Machine Learning (ICML) (2008)
- Kersting, K., van Otterlo, M., De Raedt, L.: Bellman goes relational. In: Proceedings of the International Conference on Machine Learning (ICML) (2004)
- Khardon, R.: Learning to take actions. *Machine Learning* 35(1), 57–90 (1999)
- Kochenderfer, M.J.: Evolving Hierarchical and Recursive Teleo-Reactive Programs Through Genetic Programming. In: Ryan, C., Soule, T., Keijzer, M., Tsang, E.P.K., Poli, R., Costa, E. (eds.) EuroGP 2003. LNCS, vol. 2610, pp. 83–92. Springer, Heidelberg (2003)
- Lane, T., Wilson, A.: Toward a topological theory of relational reinforcement learning for navigation tasks. In: Proceedings of the International Florida Artificial Intelligence Research Society Conference (FLAIRS) (2005)
- Lang, T., Toussaint, M.: Approximate inference for planning in stochastic relational worlds. In: Proceedings of the International Conference on Machine Learning (ICML) (2009)
- Lang, T., Toussaint, M.: Probabilistic backward and forward reasoning in stochastic relational worlds. In: Proceedings of the International Conference on Machine Learning (ICML) (2010)
- Langley, P.: Cognitive architectures and general intelligent systems. *AI Magazine* 27, 33–44 (2006)
- Lanzi, P.L.: Learning classifier systems from a reinforcement learning perspective. *Soft Computing* 6, 162–170 (2002)
- Lecoeuche, R.: Learning optimal dialogue management rules by using reinforcement learning and inductive logic programming. In: Proceedings of the North American Chapter of the Association for Computational Linguistics, NAACL (2001)
- Letia, I., Precup, D.: Developing collaborative Golog agents by reinforcement learning. In: Proceedings of the 13th IEEE International Conference on Tools with Artificial Intelli-

- gence (ICTAI 2001). IEEE Computer Society (2001)
- Levine, J., Humphreys, D.: Learning Action Strategies for Planning Domains Using Genetic Programming. In: Raidl, G.R., Cagnoni, S., Cardalda, J.J.R., Corne, D.W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C.G., Marchiori, E., Meyer, J.-A., Middendorf, M. (eds.) *EvoIASP 2003, EvoWorkshops 2003, EvoSTIM 2003, EvoROB/EvoRobot 2003, EvoCOP 2003, EvoBIO 2003, and EvoMUSART 2003*. LNCS, vol. 2611, pp. 684–695. Springer, Heidelberg (2003)
- Lison, P.: Towards relational POMDPs for adaptive dialogue management. In: *ACL 2010: Proceedings of the ACL 2010 Student Research Workshop*, pp. 7–12. Association for Computational Linguistics, Morristown (2010)
- Littman, M.L., Sutton, R.S., Singh, S.: Predictive representations of state. In: *Proceedings of the Neural Information Processing Conference (NIPS)* (2001)
- Lloyd, J.W.: *Logic for Learning: Learning Comprehensible Theories From Structured Data*. Springer, Heidelberg (2003)
- Martin, M., Geffner, H.: Learning generalized policies in planning using concept languages. In: *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR)* (2000)
- Mausam, Weld, D.S.: Solving relational MDPs with first-order machine learning. In: *Workshop on Planning under Uncertainty and Incomplete Information at ICAPS 2003* (2003)
- McCallum, R.A.: Instance-based utile distinctions for reinforcement learning with hidden state. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 387–395 (1995)
- Mellor, D.: A Learning Classifier System Approach to Relational Reinforcement Learning. In: Bacardit, J., Bernadó-Mansilla, E., Butz, M.V., Kovacs, T., Llorà, X., Takadama, K. (eds.) *IWLCS 2006 and IWLCS 2007*. LNCS (LNAI), vol. 4998, pp. 169–188. Springer, Heidelberg (2008)
- Minker, J.: *Logic-Based Artificial Intelligence*. Kluwer Academic Publishers Group, Dordrecht (2000)
- Minton, S., Carbonell, J., Knoblock, C.A., Kuokka, D.R., Etzioni, O., Gil, Y.: Explanation-based learning: A problem solving perspective. *Artificial Intelligence* 40(1-3), 63–118 (1989)
- Mooney, R.J., Califf, M.E.: Induction of first-order decision lists: Results on learning the past tense of english verbs. *Journal of Artificial Intelligence Research (JAIR)* 3, 1–24 (1995)
- Moore, A.W., Atkeson, C.G.: Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning* 13(1), 103–130 (1993)
- Morales, E.F.: Scaling up reinforcement learning with a relational representation. In: *Proceedings of the Workshop on Adaptability in Multi-Agent Systems at AORC 2003*, Sydney (2003)
- Morales, E.F.: Learning to fly by combining reinforcement learning with behavioral cloning. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 598–605 (2004)
- Moriarty, D.E., Schultz, A.C., Grefenstette, J.J.: Evolutionary algorithms for reinforcement learning. *Journal of Artificial Intelligence Research (JAIR)* 11, 241–276 (1999)
- Mourão, K., Petrick, R.P.A., Steedman, M.: Using kernel perceptrons to learn action effects for planning. In: *Proceedings of the International Conference on Cognitive Systems (CogSys)*, pp. 45–50 (2008)
- Muller, T.J., van Otterlo, M.: Evolutionary reinforcement learning in relational domains. In: *Proceedings of the 7th European Workshop on Reinforcement Learning* (2005)
- Nason, S., Laird, J.E.: Soar-RL: Integrating reinforcement learning with soar. In: *Proceedings of the Workshop on Relational Reinforcement Learning at ICML 2004* (2004)
- Nath, A., Domingos, P.: A language for relational decision theory. In: *International Workshop on Statistical Relational Learning, SRL* (2009)
- Neruda, R., Slusny, S.: Performance comparison of two reinforcement learning algorithms for small mobile robots. *International Journal of Control and Automation* 2(1), 59–68 (2009)
- Oates, T., Cohen, P.R.: Learning planning operators with conditional and probabilistic ef-

- fects. In: Planning with Incomplete Information for Robot Problems: Papers from the 1996 AAAI Spring Symposium, pp. 86–94 (1996)
- Pasula, H.M., Zettlemoyer, L.S., Kaelbling, L.P.: Learning probabilistic planning rules. In: Proceedings of the International Conference on Artificial Intelligence Planning Systems (ICAPS) (2004)
- Poole, D.: The independent choice logic for modeling multiple agents under uncertainty. *Artificial Intelligence* 94, 7–56 (1997)
- Ramon, J., Driessens, K., Croonenborghs, T.: Transfer Learning in Reinforcement Learning Problems Through Partial Policy Recycling. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) *ECML 2007. LNCS (LNAI)*, vol. 4701, pp. 699–707. Springer, Heidelberg (2007)
- Reiter, R.: *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, Cambridge (2001)
- Rodrigues, C., Gerard, P., Rouveirol, C.: On and off-policy relational reinforcement learning. In: *Late-Breaking Papers of the International Conference on Inductive Logic Programming* (2008)
- Rodrigues, C., Gérard, P., Rouveirol, C.: Incremental Learning of Relational Action Models in Noisy Environments. In: Frasconi, P., Lisi, F.A. (eds.) *ILP 2010. LNCS*, vol. 6489, pp. 206–213. Springer, Heidelberg (2011)
- Roncagliolo, S., Tadepalli, P.: Function approximation in hierarchical relational reinforcement learning. In: *Proceedings of the Workshop on Relational Reinforcement Learning at ICML* (2004)
- Russell, S.J., Norvig, P.: *Artificial Intelligence: a Modern Approach*, 2nd edn. Prentice Hall, New Jersey (2003)
- Ryan, M.R.K.: Using abstract models of behaviors to automatically generate reinforcement learning hierarchies. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 522–529 (2002)
- Saad, E.: A Logical Framework to Reinforcement Learning Using Hybrid Probabilistic Logic Programs. In: Greco, S., Lukasiewicz, T. (eds.) *SUM 2008. LNCS (LNAI)*, vol. 5291, pp. 341–355. Springer, Heidelberg (2008)
- Safaei, J., Ghassem-Sani, G.: Incremental learning of planning operators in stochastic domains. In: *Proceedings of the International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pp. 644–655 (2007)
- Sanner, S.: Simultaneous learning of structure and value in relational reinforcement learning. In: Driessens, K., Fern, A., van Otterlo, M. (eds.) *Proceedings of the ICML-2005 Workshop on Rich Representations for Reinforcement Learning* (2005)
- Sanner, S.: Online feature discovery in relational reinforcement learning. In: *Proceedings of the ICML-2006 Workshop on Open Problems in Statistical Relational Learning* (2006)
- Sanner, S., Boutilier, C.: Approximate linear programming for first-order MDPs. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)* (2005)
- Sanner, S., Boutilier, C.: Practical linear value-approximation techniques for first-order MDPs. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)* (2006)
- Sanner, S., Boutilier, C.: Approximate solution techniques for factored first-order MDPs. In: *Proceedings of the International Conference on Artificial Intelligence Planning Systems (ICAPS)* (2007)
- Sanner, S., Kersting, K.: Symbolic dynamic programming for first-order pomdps. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)* (2010)
- Schmid, U.: Inductive synthesis of functional programs: Learning domain-specific control rules and abstraction schemes. In: *Habilitationsschrift, Fakultät IV, Elektrotechnik und Informatik, Technische Universität Berlin, Germany* (2001)
- Schuermans, D., Patrascu, R.: Direct value approximation for factored MDPs. In: *Proceedings of the Neural Information Processing Conference (NIPS)* (2001)
- Shapiro, D., Langley, P.: Separating skills from preference. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 570–577 (2002)

- Simpkins, C., Bhat, S., Isbell, C.L., Mateas, M.: Adaptive Programming: Integrating Reinforcement Learning into a Programming Language. In: Proceedings of the Twenty-Third ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA (2008)
- Staney, J., Thiébaux, S.: Blocks world revisited. *Artificial Intelligence* 125, 119–153 (2001)
- Song, Z.W., Chen, X.P.: States evolution in $\Theta(\lambda)$ -learning based on logical mdps with negation. In: IEEE International Conference on Systems, Man and Cybernetics, pp. 1624–1629 (2007)
- Song, Z.W., Chen, X.P.: Agent learning in relational domains based on logical mdps with negation. *Journal of Computers* 3(9), 29–38 (2008)
- Stone, P.: Learning and multiagent reasoning for autonomous agents. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Computers and Thought Award Paper (2007)
- Stracuzzi, D.J., Asgharbeygi, N.: Transfer of knowledge structures with relational temporal difference learning. In: Proceedings of the ICML 2006 Workshop on Structural Knowledge Transfer for Machine Learning (2006)
- Sutton, R.S., Barto, A.G.: Reinforcement Learning: an Introduction. The MIT Press, Cambridge (1998)
- Sutton, R.S., McAllester, D.A., Singh, S., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: Proceedings of the Neural Information Processing Conference (NIPS), pp. 1057–1063 (2000)
- Thielscher, M.: Introduction to the Fluent Calculus. *Electronic Transactions on Artificial Intelligence* 2(3–4), 179–192 (1998)
- Thon, I., Guttman, B., van Otterlo, M., Landwehr, N., De Raedt, L.: From non-deterministic to probabilistic planning with the help of statistical relational learning. In: Workshop on Planning and Learning at ICAPS (2009)
- Torrey, L.: Relational transfer in reinforcement learning. PhD thesis, University of Wisconsin-Madison, Computer Science Department (2009)
- Torrey, L., Shavlik, J., Walker, T., Maclin, R.: Relational macros for transfer in reinforcement learning. In: Proceedings of the International Conference on Inductive Logic Programming (ILP) (2007)
- Torrey, L., Shavlik, J., Natarajan, S., Kuppli, P., Walker, T.: Transfer in reinforcement learning via markov logic networks. In: Proceedings of the AAAI-2008 Workshop on Transfer Learning for Complex Tasks (2008)
- Toussaint, M.: Probabilistic inference as a model of planned behavior. *Künstliche Intelligenz (German Artificial Intelligence Journal)* 3 (2009)
- Toussaint, M., Plath, N., Lang, T., Jetchev, N.: Integrated motor control, planning, grasping and high-level reasoning in a blocks world using probabilistic inference. In: IEEE International Conference on Robotics and Automation, ICRA (2010)
- Van den Broeck, G., Thon, I., van Otterlo, M., De Raedt, L.: DTProbLog: A decision-theoretic probabilistic prolog. In: Proceedings of the National Conference on Artificial Intelligence (AAAI) (2010)
- van Otterlo, M.: Efficient reinforcement learning using relational aggregation. In: Proceedings of the Sixth European Workshop on Reinforcement Learning, Nancy, France (EWRL-6) (2003)
- van Otterlo, M.: Reinforcement learning for relational MDPs. In: Nowé, A., Lenaerts, T., Steenhaut, K. (eds.) *Machine Learning Conference of Belgium and the Netherlands (BeNeLearn 2004)*, pp. 138–145 (2004)
- van Otterlo, M.: Intensional dynamic programming: A rosetta stone for structured dynamic programming. *Journal of Algorithms* 64, 169–191 (2009a)
- van Otterlo, M.: The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for Adaptive Sequential Decision Making under Uncertainty in First-Order and Relational Domains. IOS Press, Amsterdam (2009b)
- van Otterlo, M., De Vuyst, T.: Evolving and transferring probabilistic policies for relational reinforcement learning. In: Proceedings of the Belgium-Netherlands Artificial Intelligence Conference (BNAIC), pp. 201–208 (2009)

- van Otterlo, M., Wiering, M.A., Dastani, M., Meyer, J.J.: A characterization of sapient agents. In: Mayorga, R.V., Perlovsky, L.I. (eds.) *Toward Computational Sapience: Principles and Systems*, ch. 9. Springer, Heidelberg (2007)
- Vargas, B., Morales, E.: Solving navigation tasks with learned teleo-reactive programs, pp. 4185–4185 (2008), doi:10.1109/IROS.2008.4651240
- Vargas-Govea, B., Morales, E.: Learning Relational Grammars from Sequences of Actions. In: Bayro-Corrochano, E., Eklundh, J.-O. (eds.) *CIARP 2009*. LNCS, vol. 5856, pp. 892–900. Springer, Heidelberg (2009)
- Vere, S.A.: Induction of relational productions in the presence of background information. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 349–355 (1977)
- Walker, T., Shavlik, J., Maclin, R.: Relational reinforcement learning via sampling the space of first-order conjunctive features. In: *Proceedings of the Workshop on Relational Reinforcement Learning at ICML 2004* (2004)
- Walker, T., Torrey, L., Shavlik, J., Maclin, R.: Building relational world models for reinforcement learning. In: *Proceedings of the International Conference on Inductive Logic Programming (ILP)* (2007)
- Walsh, T.J.: Efficient learning of relational models for sequential decision making. PhD thesis, Rutgers University, Computer Science Department (2010)
- Walsh, T.J., Littman, M.L.: Efficient learning of action schemas and web-service descriptions. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)* (2008)
- Walsh, T.J., Li, L., Littman, M.L.: Transferring state abstractions between mdps. In: *ICML-2006 Workshop on Structural Knowledge Transfer for Machine Learning* (2006)
- Wang, C.: First-order markov decision processes. PhD thesis, Department of Computer Science, Tufts University, U.S.A (2007)
- Wang, C., Khardon, R.: Policy iteration for relational mdps. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)* (2007)
- Wang, C., Khardon, R.: Relational partially observable mdps. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)* (2010)
- Wang, C., Schmolze, J.: Planning with pomdps using a compact, logic-based representation. In: *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence, ICTAI* (2005)
- Wang, C., Joshi, S., Khardon, R.: First order decision diagrams for relational MDPs. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (2007)
- Wang, C., Joshi, S., Khardon, R.: First order decision diagrams for relational MDPs. *Journal of Artificial Intelligence Research (JAIR)* 31, 431–472 (2008a)
- Wang, W., Gao, Y., Chen, X., Ge, S.: Reinforcement Learning with Markov Logic Networks. In: Gelbukh, A., Morales, E.F. (eds.) *MICAI 2008*. LNCS (LNAI), vol. 5317, pp. 230–242. Springer, Heidelberg (2008b)
- Wang, X.: Learning by observation and practice: An incremental approach for planning operator acquisition. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 549–557 (1995)
- Wingate, D., Soni, V., Wolfe, B., Singh, S.: Relational knowledge with predictive state representations. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (2007)
- Wooldridge, M.: *An introduction to MultiAgent Systems*. John Wiley & Sons Ltd., West Sussex (2002)
- Wu, J.H., Givan, R.: Discovering relational domain features for probabilistic planning. In: *Proceedings of the International Conference on Artificial Intelligence Planning Systems (ICAPS)* (2007)
- Wu, K., Yang, Q., Jiang, Y.: ARMS: Action-relation modelling system for learning action models. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)* (2005)
- Xu, J.Z., Laird, J.E.: Instance-based online learning of deterministic relational action models. In: *Proceedings of the International Conference on Machine Learning (ICML)* (2010)
- Yoon, S.W., Fern, A., Givan, R.: Inductive policy selection for first-order MDPs. In: *Proceed-*

- ings of the Conference on Uncertainty in Artificial Intelligence (UAI) (2002)
- Zettlemoyer, L.S., Pasula, H.M., Kaelbling, L.P.: Learning planning rules in noisy stochastic worlds. In: Proceedings of the National Conference on Artificial Intelligence (AAAI) (2005)
- Zhao, H., Doshi, P.: Haley: A hierarchical framework for logical composition of web services. In: Proceedings of the International Conference on Web Services (ICWS), pp. 312–319 (2007)
- Zhuo, H., Li, L., Bian, R., Wan, H.: Requirement Specification Based on Action Model Learning. In: Huang, D.-S., Heutte, L., Loog, M. (eds.) ICIC 2007. LNCS, vol. 4681, pp. 565–574. Springer, Heidelberg (2007)

层次式技术

Bernhard Hengst

摘要

层次分解把复杂问题分解为较小的一组相关问题进行处理。这些较小的问题可以分别解决，重新整合其结果后可以形成原始问题的解。众所周知，简单应用强化学习技术无法处理复杂领域问题。本章介绍层次式强化学习技术，从而有望将强化学习问题规约到能够操作的长度。层次式强化学习（Hierarchical Reinforcement Learning，HRL）依赖于寻找良好的、可重用、可扩展的技术，同时这些技术也能够提供对状态进行抽象的机会。强化学习方法扩展到能够处理经过层次式分解原始问题而形成的抽象状态，从而减少计算复杂度。我们使用四房间任务作为示例讲解各种概念和技术，包括能够自动从领域交互中学习层次结构的算法。

9.1 简介

人工智能（AI）是关于怎样构造能够理性动作的智能主体的学科。当学习器能够在感知序列下最大化某种性能指标时，则能够理性动作 [Russell and Norvig, 1995]。从学习器视角去看的规划和控制理论也属于本类问题。粗看起来，强化学习可能是解决人工通用智能（Artificial General Intelligence, AGI）的一种相当诱人的方法 [Hutter, 2007]。然而这种看法只在原则上成立，强化学习经常被“维度灾难（curse of dimensionality）”问题困扰。“维度灾难”是由 [Bellman, 1961] 提出的概念，指在描述问题的变量和维度增加时状态空间以指数级增长的问题。贝尔曼指出单纯枚举无法解决任何真正具有意义的问题。由于不太可能仅仅依靠一组变量就能描述复杂问题，看来我们遭遇了困境。

293

幸运的是，真实世界的特点是高度结构化并具备多重约束，其中大部分与其他部分互相独立。如果不存在结构，则不可能解决任意规模的复杂问题 [Russell and Norvig, 1995]。内在结构能够大幅度减少由单纯枚举而产生的简单状态空间。例如，如果针对两个变量的转换和奖励函数互相独立，那么强化学习主体只需要处理变量状态空间的和，而不是状态空间的积。

强化学习关注能够被动作和状态表征的问题，把解决问题推广到具有时域动态性的问题。因此，我们可以用任务、子问题和子任务来指代强化学习问题。

本章关注利用层次式结构降低复杂度解决不如此则无法攻克的强化学习问题。

1. 层次

从经验来看，绝大多数来自大自然的复杂系统表现出层次结构。这里“层次”指的是系统由一系列相关子系统构成，这些子系统拥有自身的下一级系统，以此类推。从埃及时代开始，人类社会一直在使用层次组织解决复杂任务。层次式系统具有“几乎可分解”的特点，

意味着单元内部的连接要强于单元间的连接 [Simon 1996]。这种“几乎可分解”的特点有助于简化系统的动作和描述。

[Polya, 1945] 提出了一种解决问题的启发式方法，即“分解和重组”，今天一般称之为“分治法”。许多大型问题拥有层次式结构，允许将问题分解为子问题。这些子问题规模较小，因此可以更加容易地解决。对于问题的解进行重组后，就可以形成原始大型问题的解。通过改进学习和执行过程的时间和空间复杂度，以上分解的方法可以大幅度提升问题求解的效率。

层次化能够简化问题求解和提升解决效率的另外一个原因是，经常会有相似的问题需要在不同环境下求解。如果强化学习主体不知道内在的结构，则只好在不同环境下对同一任务多次学习，然而更好的方法显然是只学习一次并在之后重用学习结果。程序员使用函数调用和子程序来实现相似的目标——避免重复的代码片段。在许多方面，层次式强化学习 (Hierarchical Reinforcement Learning, HRL) 对问题进行结构化的过程非常类似于计算机编程，这里子程序对应于更高层次强化学习问题的子任务。正如主函数调用子程序，更高层次的强化学习问题激活子任务。

294

2. 四房间问题

在本章中，我们使用一个简单的四房间任务作为实例讲解相关概念。图 9.1a 给出学习器视角的强化学习。学习器通过一个感知-动作的循环与环境交互，在每个时间步接受一个奖励信号作为输入的一部分。

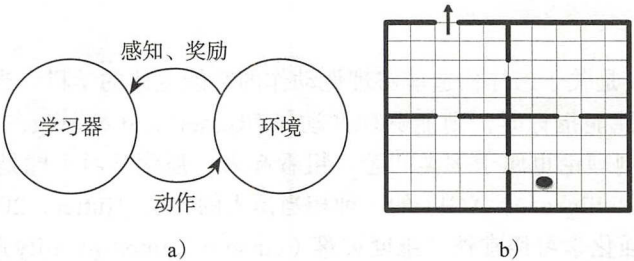


图 9.1 a) 学习器视角的强化学习；b) 四房间任务，以黑色实心椭圆表示的学习器位于其中一个房间

在这个例子中，以黑色实心椭圆表示的学习器位于一座拥有四个房间的住所内部东南角房间（见图 9.1b）的某个格点。这些房间由开放的过道连接，西北角的房间还有一个走向住所外部的过道。在每个时间步长上，学习器执行一个动作并接受传感器的观测结果以及来自环境的奖励信号。

每个单元格点代表学习器的一个可能位置，可以被房间和坐标唯一表征。本例中的房间都有相同的尺度，各个房间中的相似位置由相同标志符描述。如果学习器能够感知它所在的房间和房间内的位置，则认为环境是完全可观察的 (fully observable)。学习器在每个时间步长上可以在东、南、西、北中的一个方向上移动一格，并且会收到一个 -1 的奖励。目标是通过最短路径离开房间。我们假定动作是随机的，当采取动作时，学习器以 80% 的概率在理想的方向移动，并以 20% 的概率保持原地。如果学习器撞到墙，则保持不动。

295

3. 层次式强化学习

我们的 HBL 技术首先把一个完备定义的强化学习问题表征为第 1 章描述的马尔可夫决策过程 (Markov Decision Process, MDP)。读者可以很容易验证四房间任务是满足要求的强

化学习问题。这里我们提供一个怎样把 HRL 方法用于四房间任务的起始的、直观的描述。

如果我们能找到离开房间的策略，例如离开北侧的过道，因为房间都一样，那么我们就可以可以在所有房间重用这一策略。从北侧过道离开房间就是一个简化过的强化学习问题，但同时又继承了原始问题的房间内位置状态、动作、转换函数和奖励函数。我们接下来解决两个简化的强化学习问题，一个是从北方离开房间的策略，另一个是从西部走道离开房间的策略。

我们也表述和解决较高层次的、只使用四个房间状态的强化学习问题。在每一个房间的状态上，我们允许选择执行在此前房间学到的离开房间的策略。对于较高层次的问题来说，这些策略被认为是可以时域扩展 (temporally extended) 的动作，之所以这样说，是由于它们一旦被激活后，经常会在多个时间步长上持续执行直至学习器离开房间。在这个阶段，我们简单地离开房间的动作制定一个奖励值 -1。此后会看到，强化学习可以通过半马尔可夫决策过程 (Semi-Markov Decision Process, SMDP) 推广到时域扩展动作上。

一旦学习完成，较高层次的离开房屋的策略执行将决定离开当前房间的策略。此时控制权被转换到离开房间的子任务，从而使得学习器能够从选定的走道离开房间。离开房间时，子任务中止，控制权返回较高层次并选择下一个离开房间的动作直至离开整个房屋。

上面的例子隐藏了许多 HRL 需要处理的细节，例如安全的状态抽象、恰当的计算累计子任务奖励、优化求解以及描述甚至学习层次式结构自身。下一节我们讨论这些问题并回顾 HRL 技术。

9.2 背景

本节介绍对于理解层次式强化学习很重要的一些概念。一般来说，时域扩展动作和相关半马尔可夫决策过程表述是关键的成分。我们也会讨论减少问题规模和求解优化性的相关问题。

[296]

9.2.1 抽象动作

HRL 技术会使用在多个时间步长上反复执行的动作。这些时域扩展动作 (或抽象动作) 隐藏了从被激活到中止时间之内的多个时间步长上状态转换和奖励的细节。上述四房间任务中离开房间的动作就是这样的例子。离开一个房间可以涉及多个单时间步长上的动作，这些动作包括导航至过道然后穿越过道。

包括人工智能、机器人和控制工程的许多领域都会使用抽象动作的概念。这很像计算机科学中的“宏 (macro)”，也就是把一系列指令编码为一个单一的命令。在规划任务中，宏能够分解并解决问题 (请参考求解数字推盘 (Fifteen Puzzle) 问题和魔方的例子 [Korf, 1985])。

MDP 背景下的抽象动作对宏的概念进行了扩展，允许把更加初级的有一个抽象动作组成的步骤模型化为随机转换函数并使用随机策略 (参阅第 1 章)。抽象动作可以针对一个较小的马尔可夫决策问题执行策略。随机性可以表现在多个方面。当执行抽象动作时，随机转换函数可能造成不确定性的状态访问序列。即使奖励函数是确定性的，奖励信号的顺序同样可能根据状态访问序列而变化。最后，完成一个抽象动作的时间也可能改变。实际上，具有确定后果的抽象动作只能看作经典的宏操作。

我们可以从四房间问题检查抽象动作的特性。在这个问题中，当一个抽象动作被激活时，如果有 20% 的机会学习器原地不动，这并不意味着事先就可以确定还需要多少个时间

步长就可以离开房间。

抽象动作可以被推广到连续时间域 [Puterman, 1994]。但是, 本章聚焦于离散时间域问题。在一个时间步长上结束普通动作是抽象动作的特例, 我们将其称为简单动作 (primitive action)。

9.2.2 半马尔可夫决策问题

我们现在把第 1 章的 MDP 扩展到包含抽象动作。这样的 MDP 称为半马尔可夫决策问题 (SMDP) [Puterman, 1994]。由于抽象动作可以在随机个数时间步长上完成, 我们需要引入另一个变量表征抽象动作的完成时间。

297 针对从状态 s 开始、在状态 s' 结束的抽象动作 a , 我们将其执行的时间步数记作随机变量 N ($N \geq 1$)。由状态转换概率函数和期望奖励函数定义的 SMDP 模型现在包含了随机变量 N^\ominus 。

转换函数 $T: S \times A \times S \times N \rightarrow [0,1]$ 给出抽象动作 a 从初始状态 s 开始经过 N 个时间步长后在状态 s' 中止的概率。

$$T(s, a, s', N) = \Pr\{s_{t+N} = s' \mid s_t = s, a_t = a\} \quad (9.1)$$

针对抽象动作的奖励函数在执行过程中累积单步的奖励, 而累积的方式依赖于 SMDP 内嵌的性能评估方式。常见的一种性能评估方法是用一个常数阻尼系数 γ 来最大化未来折扣奖励之和。奖励函数 $R: S \times A \times S \times N \rightarrow \mathbb{R}$ 给出抽象动作 a 从初始状态 s 开始经过 N 个时间步长后在状态 s' 中止的期望折扣奖励之和, 其形式如下:

$$R(s, a, s', N) = E \left\{ \sum_{n=0}^{N-1} \gamma^n r_{t+n} \mid s_t = s, a_t = a, s_{t+N} = s' \right\} \quad (9.2)$$

第 1 章中 MDP 的价值函数和贝尔曼备份方程也可以推广到 SMDP。状态 s 对策略 π 的价值记作 $V^\pi(s)$, 就是在时间 t 状态 s 下根据策略 π 执行抽象动作的价值为^①:

$$V^\pi(s) = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s \right\}$$

如果在状态 s 执行的抽象动作是 $\pi(s)$, 持续 N 步结束, 我们可以把价值函数写作两个级数, 即最初的 N 个步骤的积累奖励之和以及奖励序列的剩余级数:

$$V^\pi(s) = E_\pi \left\{ (r_t + \gamma r_{t+1} + \cdots + \gamma^{N-1} r_{t+N-1}) + (\gamma^N r_{t+N} + \cdots) \mid s_t = s \right\}$$

在公式 (9.1) 定义的概率上针对 s' 和 N 计算数学期望, 替换到公式 (9.2) 的抽象动作 $\pi(s)$ 的 N 步奖励, 并且可以发现第二个级数就是从 s' 开始折扣 N 步的价值函数, 上式可以写为:

$$298 \quad V^\pi(s) = \sum_{s', N} T(s, \pi(s), s', N) [R(s, \pi(s), s', N) + \gamma^N V^\pi(s')]]$$

SMDP 的最优价值函数 (用 $*$ 标记) 也与 MDP 类似, 针对 s' 和 N 取累加和的形式:

$$V^*(s) = \max_a \sum_{s', N} T(s, a, s', N) [R(s, a, s', N) + \gamma^N V^*(s')]$$

类似的, 可以为 SMDP 策略写出 Q 抽象动作价值函数。 $Q\pi(s, a)$ 定义为在状态 s 采取抽象动作 a , 并在此后遵循策略 π 。因此:

① SMDP 的这种表述是在 [Sutton et al, 1999] 和 [Dietterich, 2000] 基础上修改成与第 1 章符号一致的形式。

② 注意, 我们重载了函数 π , $\pi(s, a)$ 是在状态 s 选择抽象动作 a 的概率, $\pi(s)$ 是在状态 s 根据确定性策略 π 选择的动作。

$$Q^{\pi}(s, a) = \sum_{s', N} T(s, a, s', N) [R(s, a, s', N) + \gamma^N Q^{\pi}(s', \pi(s'))] \quad (9.3)$$

SMDP 最优价值函数为：

$$Q^*(s, a) = \sum_{s', N} T(s, a, s', N) [R(s, a, s', N) + \gamma^N V^*(s')] \\ \text{其中 } V^*(s') = \max_a Q^*(s', a')$$

对于保证中止的问题，阻尼系数 γ 可以设为 1。在这种情况下上式中的时间步数 N 可以被边际化 (marginalised)，因此只与 s 有关。这样，以上公式与 MDP 相应公式类似，只是把简单奖励的期望替换为指导执行中止时的奖励和的期望。

所有第 1 章中用来解决针对简单动作的马尔可夫决策过程的强化学习方法对抽象动作依然有效。由于简单动作只是抽象动作的特例，我们把它们也包括在抽象动作之内。

读者可能会想知道引入抽象动作是否有意义，实际上我们为问题引入了额外的动作从而增加了复杂度。本章后续部分会解释抽象动作怎样利用问题的结构从而减少存储容量要求并加快学习速度。

在一个类似于图 9.1 四房间问题的例子中，[Sutton et al, 1999] 解释了抽象动作的引入允许学习器通过逐个房间搜索而不是逐个位置搜索的方法显著提高逐学习速度^①。当目标不在现有抽象动作容易到达的位置时，可以使用作为特例的简单动作并加速某些问题的学习。例如，只有离开房间的抽象动作时，我们可能无法到达房间中部的某个位置。

除非我们引入其他抽象动作，当进入包含目标状态的房间时我们仍然需要简单动作。尽管包含简单动作能够保证向全局最优策略收敛，也同样可能导致额外的学习工作。由于价值函数可以在状态空间中后退很长的距离，同时包括了保证全局最优收敛的简单动作，强化学习过程可以被加速，但是这些额外动作的引入同样增加了所需的存储和探索空间。

[299]

9.2.3 结构

抽象动作和 SMDP 天然导致层次结构。通过恰当的抽象动作本身，相对于通过简单动作解决问题，我们可以更加容易地学习策略。这是因为抽象动作可以略去很大一部分中止于小的状态集合的状态空间。在四房间任务中，我们已经看到离开房间的抽象动作可以把问题的状态空间减少到只有房间（而没有房间内部）的状态。

抽象动作本身也可以针对较小的 SMDP 或 MDP，这样就建立了一个层次，其中高层次的父任务调用子任务作为抽象动作。

1. 任务层次

SMDP 中的复杂关系导致了任务层次 [Dietterich, 2000]。任务层次是由子任务构成的有向无环图，其中根节点是以抽象动作形式激活其子节点 SMDP 策略的顶层 SMPD。子节点策略以递归方式激活其子任务，直至只拥有简单动作的子任务。设计者需要谨慎定义在每个父任务状态空间中可以使用的共有抽象动作，并且指定每个子任务的激活和中止状态。

图 9.2 给出 9.1 节讨论过的四房间问题的任务层次。两个低层子任务离开一个典型房间的 MDP，分别学习从北侧和西侧离开房间的策略。箭头表示向中止状态的转换。状态、动作、转换和奖励均从初始 MDP 问题继承而来。高层次问题 (SMDP) 仅有表示房间的四个状态组成，任何子任务（离开房间的动作）可以在任何房间激活。

① 在本例中，在 9.3.1 节定义的抽象动作以选项形式表示。

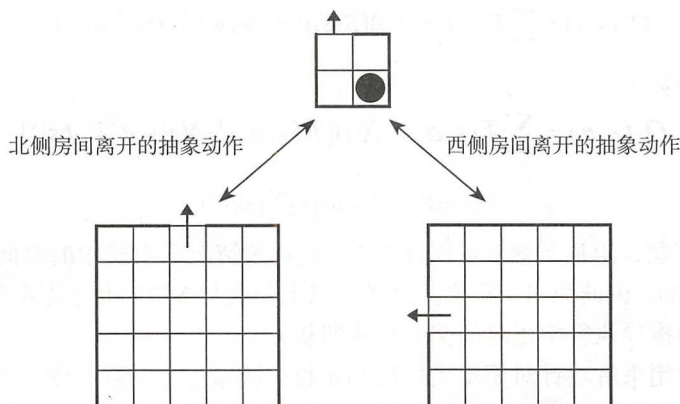


图 9.2 分解图 9.1 中四房间任务的任务层次。两个低层子任务是通用的离开房间的抽象动作，分别负责从北侧和西侧离开房间

2. 部分程序

手工编制任务层次其实是一种编程过程，其中设计者需要使用背景知识从而利用问题的层次结构。不是所有的子任务策略均需要学习。程序员已经拥有良好子任务策略的情况下，这些策略可以被直接编码为随机有限状态机的形式。

由于计算机本身是有限状态机，可以对随机有限状态机进行近似，任务层次可以被有效表征为利用已有程序结构的部分程序。我们将在 9.3.2 节详细讨论这种 HRL 技术。

9.2.4 状态抽象

拆解一个大 MDP 问题的在于可能找到抽象状态的机会从而减少问题复杂度。抽象状态空间通常小于原始 MDP 问题的状态空间。广义上看，在两类条件下可以引入状态抽象 [Dietterich, 2000]:

- 可以去除不相关变量。
- 抽象动作提供使学习器走向较小状态集合的捷径（或者“漏斗”）。

1. 去除不相关变量

当接收到冗余信息时，强化学习算法会针对所有冗余状态学习到同样的价值函数或策略。例如，探索一个红色的房间可能完全等同于蓝色的房间，但是价值函数和策略会把每一个（房间中的位置、颜色）组合作为一个单独状态。如果颜色对探索没有效果，则可以通过删除颜色变量简化问题。

更概括地说，如果我们找到把状态空间分为 m 个子任务的拆分，并能够满足所有单个子空间内的状态转换具有相同转换概率和向其他子空间转换的期望奖励，我们就可以把子任务简化到每一个拆分空间对应一个状态。对简化子任务的解也是原始问题的解，这就是最小化 MDP 模型的随机互模拟一致性（stochastic bisimulation homogeneity）概念 [Dean and Givan, 1997]。求解 MDP 的计算复杂度，一般是 $|S|$ 的多项式函数，可以通过拆分的粗粒度化进行简化。状态抽象的效果可以非常明显。以散步子任务为例，该任务基本上与地理环境、服饰以及周边物体相互独立。如果不能抽象散步子任务，我们可能不得不在每次以上变量改变数值时都重新学习散步过程。

用形式化语言来说，如果 $P = \{B_1, \dots, B_n\}$ 是一个 SMDP 问题状态空间的一部分，并且对于

每个 $B_i, B_j \in P, a \in A, p, q \in B_i$, 执行至中止的时间步数为 N , 那么当且仅当以下条件成立:

$$\begin{aligned}\sum_{r \in B_j} T(p, a, r, N) &= \sum_{r \in B_j} T(q, a, r, N) \\ \sum_{r \in B_j} R(p, a, r, N) &= \sum_{r \in B_j} R(q, a, r, N)\end{aligned}$$

则子任务可以最小化为单一子任务, 其中 S 被 P 代替:

$$\begin{aligned}T(B_i, a, B_j, N) &= \sum_{r \in B_j} T(p, a, r, N) \\ R(B_i, a, B_j, N) &= \sum_{r \in B_j} R(p, a, r, N)\end{aligned}$$

四房间任务中离开房间的抽象动作是模型最小化 (minimisation) 的例子。对这些子任务来说, 具体的房间无关紧要, 因此房间变量可以被去掉。在这个例子中, 划分整个状态空间而成的区块由拥有相同房间内位置价值的状态组成。

[Boutilier et al, 1995] 使用了这种状态抽象方法从而利用动态贝叶斯网络变量间的独立性。该方法被 [Dietterich, 2000] 引入到 MAXQ 图的最大节点和叶子节点无关性 (9.3.3 节)。^[302] [Ravindran and Barto, 2003] 将这种形式的模型最小化延伸到隐藏于代数形式化态射 (morphisms) 的状态-动作的对称性。最小化模型的状态条件是一个同态域 (homomorphic image), 因此转换和奖励可以交换。

对于任务层次中的一些子任务来说, 有些状态可以是不可达的。这种“遮蔽 (shielding)”条件是另一种形式的消除无关变量, 这种情况由任务图结构而产生。

2. 漏斗现象

漏斗 [Dietterich, 2000] 是一种状态抽象, 其中抽象动作将环境从大量的初始状态转移到少量的结果状态。例如, [Forestier and Varaiya, 1978] 中提出的工厂控制以及 [Dean and Lin, 1995] 中利用了漏斗的效果来减小 MDP 的大小。

在四房间任务中也可以观察到漏斗现象。离开房间的抽象动作把学习器从房间中的任何位置移动到对应于相应过道之外的状态。漏斗现象允许四房间任务在根节点把状态抽象到只有四个状态, 这是由于不管在任何房间的起始位置如何, 抽象动作总可以把学习器移动到另一个房间状态。

9.2.5 价值函数分解

图 9.2 中四房间任务的任务层次拥有两个成功的高层次策略, 可以找到从东南房间起始位置到房子之外的路径, 即按照北-西-北或西-北-北的顺序依次离开房间, 后者是较短的路径, 但是 9.1 节的简单层次增强式学习器无法区分距离的差距。

现在需要一种在任务层次体系上针对整个问题的价值函数分解的方法。有了这种拆解, 我们就可以在较高层次进行决策时考虑子任务的奖励。

为了理解这一点, 现在考虑学习器在图 9.1 的四房间任务中需要做的第一个决策。决定是激活离开北边房间还是离开西边房间的抽象动作并不仅仅依赖于学习器的房间状态 (在本例中为位于南边的房间), 也依赖于当前房间中的当前位置。在本例中, 我们应该把目标选择为最近的过道, 这是因为只要我们离开房间, 则从任意过道到离开整座房屋的距离是一样的。我们既需要所在的房间状态, 也需要房间内位置, 从而决定最佳动作。

现在的问题是怎样根据任务层次拆解原始的价值函数, 从而可以在每个状态选择最优动

作。9.3.3 节使用 MAXQ 技术 [Dietterich, 2000] 解决这一问题, 即为每一个子任务使用二分价值函数的方法。分割后的两部分分别处理中止抽象动作和中止子任务的价值。

[Andre and Russell, 2002] 引入了一种三分价值函数的方法, 还包含了当子任务结束时解决整个问题的价值。这个方法的优势在于整个问题背景可以被考虑进来, 从而做到针对层次的优化 (见 9.2.6 节), 而代价通常在于较低的状态抽象程度。以上分解的好处在于提供了状态抽象的机会。

9.2.6 优化

通过本书第 1 章我们已经熟悉 MDP 问题最优策略的概念。然而, 一般来说 HRL 不能保证分解后的问题也必然产生最优解。能否最优依赖于问题本身和分解的质量, 其中分解质量体现在可用的抽象动作和任务层次结构或部分问题。

1. 层次式最优

层次式最优 (hierarchically optimal) 策略在任务层次定义的约束的前提下最大化整体价值函数。为了解释这个概念, 假定四房间任务的学习器有 70% 的概率在理想的方向上移动, 而在其他三个方向上也分别有 10% 的概率。执行图 9.2 所示的层次最优策略并不见得是全局最优。顶层策略将选择离开西边房间的动作从而移动到最近的过道。即使主体由于随机漂移找到最近的北边过道, 它仍然会固执地按照离开西边房间的策略坚持从西边过道离开。这显然不是最优解。“扁平化 (flat)” 的执行完全优化的强化学习器此时应该改变策略去通过最近的过道尝试西南房间。

如果我们把抽象任务更改为既可以从西边过道也可以从北边过道离开房间, 则任务层次可以改造为支持最优解。

2. 递归式最优

[Dietterich, 2000] 引入了另一种最优形式——递归式最优。在这种优化形式下, 到达目标中止状态的子任务策略是环境无关的, 即忽略父任务的需求。这种表述方式的优点是子任务可以在不同环境下重用, 但在各个特定场合不一定最优。

递归最优解不可能比层次最优解更好。事实上, 层次最优解可以任意劣于全局最优解, 而递归最优解又可以任意劣于层次最优解。解的优化程度依赖于任务层次的设计。

3. 层次贪婪式最优

从上述内容可以看出, MDP 的随机性本质意味着抽象动作的适当程度可以在动作激活后发生改变, 并且另一个动作可能由于内在的随机转换而变成更优化的解 [Hauskrecht et al, 1998]。向中止状态前进的子任务策略可以变得不够优化。通过不断中止子任务可以选择更好的子任务。Dietterich 把这种“轮询”过程称为层次贪婪式执行。虽然这种方式必然不劣于甚至可以显著优于层次式最优或递归式最优, 但仍未提供全局最优的保证。

[Hauskrecht et al, 1998] 讨论了保证最优的分解和求解技术, 但是, 除非 MDP 可以分解为微弱耦合的更小的 MDP, 计算复杂度并不一定能够降低。

9.3 层次式强化学习技术

层次式强化学习 (HRL) 仍然是活跃的研究领域, 我们现在简要综述相关工作, 请参阅层次式强化学习的研究进展综述 [Barto and Mahadevan 2003]、层次式决策及并行、多学习器和部分可观察技术的综述 [Si et al 2004] 以及从另一个角度综述 HRL 发展动机的综述

[Ryan 2004]。

历史的视角

[Ashby 1956] 讨论了以一系列步骤放大型系统的控制问题，并把这些层次式控制系统称为“超稳定”。在更高层次上 HRL 可以视为一种门控机制，学习怎样在较低层次切换到适当和更具反应性的动作。[Ashby, 1952] 针对处理递归情况的学习器提出了这样的门控机制。

包容体系结构 (subsumption architecture) [Brooks, 1990] 以类似方式工作，该方法把复杂动作拆分为若干简单任务并组织为层次，其中层次越高抽象程度也越高。每一层的目标都包含其子层次的目标。例如，搜寻食物任务的父任务包含回避障碍物并在机器人检测到障碍物的时候激活。

[Watkins, 1989] 讨论了使用由各层耦合马尔可夫决策问题组成的层次控制的可行性。他的例子在类似 18 世纪航海者的顶层问题上提供了一种门控机制，指导舵手应该怎样航行。[Singh, 1992] 开发出了称为层次 DYNA (H-DYNA) 的门控机制。DYNA 是一种强化学习方法，能够在构造奖励和状态转换函数之后利用实际和仿真经验。H-DYNA 首先学习诸如航行到特定位置的简单任务，然后这些任务被处理为更高层次的抽象动作。

“封建式 (feudal)” 强化学习算法 [Dayan and Hinton, 1992] 强调 HRL 的另一个良好特性——状态抽象。称之为“信息隐藏”，就是说决策模型应该通过研制控制层次向上粗粒化的过程构造。

在目标层次距离 (Hierarchical Distance to Goal, HDG) 算法中，[Kaelbling, 1993] 引入了从朝向目标路径的距离模块上组合价值函数的方法。HDG 模型建立于依靠路边的导航问题之上，其思想是学习并存储与相邻路标的局部距离以及每个路标区域内两个位置之间的距离，另一个函数存储路标之间的最短距离（在该距离已经可以从前述局部距离中获得之后）。HDG 算法的目标是朝向任务目标的下一个最近的路标，并使用距离函数引导简单动作。因此，学习器很少真正走过路标，而是用路标作为走向目标的参考点。该算法为 MAXQ 价值函数分解（将在 9.3.3 节讨论）和改进层次贪婪式最优的价值函数提供了灵感（见 9.2.6 节）。[305]

[Moore et al, 1999] 使用“机场层次 (airport-hierarchy)” 算法扩展了 HDG 技术。其目标是找到从一个起始状态到一个终点状态的最优移动策略，其中起始状态和终点状态都来自于全部状态的集合。这种多目标问题要求具有 $|S|^2$ 个状态的 MDP。通过学习一组到达目标状态下越来越小区域的抽象动作，该算法能够近似通向任意目标状态的最优策略，在最好情况下其访问的状态数为 $N \log N$ 。

在不同层次混合其他学习和规划方法的混合学习技术中也可以使用抽象动作。例如，[Ryan and Reid, 2000] 使用称作 RL-TOP 的混合方法构造任务层次。在 RL-TOP 方法中，在抽象层次上使用基于 teleo-reactive 的规划技术 [Nilsson, 1994] 激活反映式的时间延伸规划算子，其中的算法利用强化学习去达到作为子目标的后置条件。

HRL 由三种范式主宰，即：选项，抽象动作的形式化；HAMQ，部分规划技术；MAXQ，包括状态抽象的价值函数分解。

9.3.1 选项

选项 (option) 是抽象动作的一种形式化方式 [Sutton et al 1999][⊖]。

⊖ 抽象动作也可以以略微不同的方式形式化，一个替代方式是使用后述的 HEXQ 算法。

定义 9.3.1 选项 (相关于 $\text{MDP} \langle S, A, T, R \rangle$) 是一个三元组 $\langle I, \pi, \beta \rangle$, 其中 $I \subseteq S$ 是初始集合, $\pi: S \times A \rightarrow [0, 1]$ 是一种策略, $\beta: S^+ \rightarrow [0, 1]$ 是中止条件。

如果 $s \in I$, 则 MDP 状态 $s \in S$ 可以选择选项。这样使得选项的初始化局限于 S 的子集。一旦激活, 选项采用由随机策略 π 指定的动作。选项根据函数 β 随机中止, 其中 β 指定在每个状态中止的概率。许多任务是插曲性 (episodic) 的, 即最终必将中止。在定义 MDP 状态空间时包含抽象中止状态, 我们把包含抽象中止状态的状态空间记作 S^+ 。如果选项在状态 s , 则以概率 $\beta(s)$ 中止, 否则以概率 $\pi(s, a)$ 选择下一个动作 a 继续执行。

当选项策略和中止只依赖于当前状态 s , 则称为马尔可夫选项。我们也许希望让某个策略也依靠当前状态 s 之外的信息。例如, 如果希望选项能够在一定时间步数后超时中断, 则需要在状态空间增加一个计数器。通常选项和中止可以依赖于从选项启动开始整个状态、动作和奖励序列的历史。这类选项称为半马尔可夫选项。我们后面将讨论通过在 MDP 上叠加随机有限状态机 (如程序) 形式化选项的例子。

很容易看出一个简单动作就是一个选项。简单动作 a 等同于选项 $\langle I = s, \pi(s, a) = 1.0, \beta(s) = 1 \rangle$, 其中 $s \in S$ 。可以整合选项集合和抽象动作集合, 用集合 A 定义其并集。

就像可以有针对简单动作的策略, 我们可以定义针对选项的策略。选项策略 $\pi: S \times A \rightarrow [0, 1]$ 是一个以概率 $\pi(s, a)$, $s \in S, a \in A$ 选择选项的函数。由于选项选择动作, 而动作就是特殊类型的选项, 所以选项也可以选择其他选项, 这样我们可以形成任意深度的层次结构,

9.3.2 HAMQ 学习

如果已知一小组手工确定并用于求解 MDP 的抽象动作, 则学习策略的过程可以比只有简单动作的情况轻松得多。这是因为抽象动作略过了状态空间中那些中止于小子集的一大部分状态。原始 MDP 可以变小, 因为现在只需要在缩减后的状态空间上学习抽象动作。

使用层次抽象机 (Hierarchy of Abstract Machine, HAM) 技术时, 设计师通过称为“抽象机 (abstract machine)”并与 MDP 协同工作的随机有限状态机制定抽象动作 [Parr and Russell, 1997]。该技术显式定义抽象动作, 从而允许使用者引入更加通用的、以部分规划形式的背景知识, 拥有多个层次的表示能力。

抽象机是一个三元组 $\langle \mu, I, \delta \rangle$, 其中 μ 是一组有限状态集合, I 是从 MDP 状态到机器状态的随机函数, 决定起始机器状态, δ 是把机器状态和 MDP 状态映射到下一机器状态的随机函数。机器状态有若干种类, 其中动作态 (action-state) 制定求解给定 MDP 状态时需要采取的动作, 调用态 (call-state) 以子程序方式执行另一个机器, 选择态 (choice-state) 以不确定方式选择下一状态, 停止态 (halt-state) 中止机器运行。

抽象机和 MDP 的并行动作形成了一个离散时间的高层 SMDP。选择态可以启动抽象动作。抽象机的动作态和选择态产生针对某个抽象策略的一组动作。如果在当前执行中止之前遇到其他选择态, 则相当于选择另一个抽象动作, 即生产了另一层的层次。抽象动作由停止态中止。审慎选择 HAM 可以减少原始 MDP 及其选择点的状态集合。

我们以四房间问题为例解释 HAM 的操作。图 9.3 的抽象机提供从西边或北边离开房间的选择。在任一房间中, 策略都是把学习器移动到墙, 然后沿墙随机游走直至找到过道。选择态之间的简单奖励进行累加。在这个例子中, 我们假定学习器的初始位置如图 9.1 所示。只有五个原始 MDP 的状态仍然是 SMDP 的状态, 这些就是学习器的初始状态和过道另一边抽象机进入选择态的状态。强化学习方法以传统方式更新这五个状态的价值函数, 并积累奖

励直至最后一个选择态。

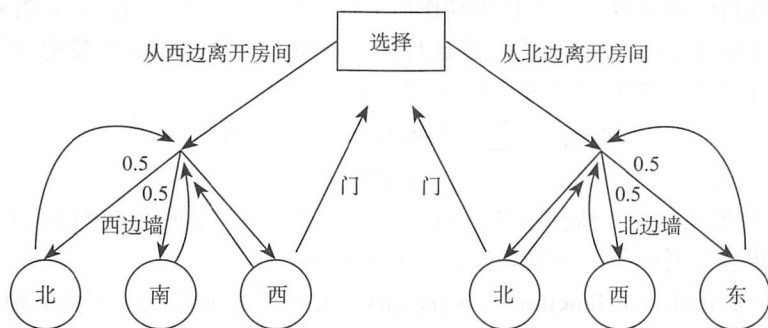


图 9.3 实现向西和向北方向离开房屋操作的抽象机。其选择限于两个抽象动作，一个是从西边过道离开房屋，另一个是通过北边的过道。如果选择西边过道，抽象动作反复选择向西的简单动作直至穿过道中止或碰到西边的墙。遇到西边墙后，随机采用向北或南的简单动作直至看不到墙为止，即到达过道，随之可以穿过道中止

求解 SMDP 产生使学习器在抽象机规划约束下离开房屋的最优策略。最好的策略由三个抽象动作组成，即依次向西、向北、再向北离开房间。这个例子并不是全局最优策略，这是因为沿墙随机游走找到过道的办法不够有效。工程师和控制理论家们预期 HAM 能够产生实现特定动作的良好控制器。HAM 也是一种部分描述过程知识从而把 MDP 转换为化简的 SMDP 的方法。

308

可编程 HRL

在最一般情况下，HAM 可以是一段程序，执行学习器完整感知-动作历史的任何可计算映射 [Parr, 1998]。

通过为 HRL 引入更具表示能力的学习器设计语言——可编程 HAM (PHAM) [Andre and Russell, 2000] 和一种基于 Lisp 的高层次部分编程语言 (ALisp) [Andre and Russell, 2002] 扩展了 HAM 技术。

Golog 是一种针对学习器的逻辑编程语言，允许学习器使用情境微积分 (situation calculus) [Levesque et al, 1997] 进行动作、目标、感知和其他学习器相关的推理。该语言后来通过嵌入 MDP 扩展为部分程序，典型例子包括决策理论 Golog (Decision Theoretic Golog, DTGolog) [Boutilier et al, 2000] 和使用选项框架的 Readylog [Ferrein and Lakemeyer, 2008]。

在以上工作中，部分程序使得使用者能够通过特殊的选择点例程 (choice-point routine) 提供关于问题机构的背景知识，这些选择点例程为学习器实现非确定性动作从而从经验中学习最佳动作。这类程序利用了程序语言的表达能力，从而更加简洁地定义 (和求解) SMDP。

9.3.3 MAXQ

MAXQ 是 HRL 的一种技术，其中价值函数在任务层次上分解 [Dietterich, 2000]。该技术可以形成简洁的价值函数表征，并使得子任务与环境无关 (可移植)。

MAXQ 的抽象动作通过把子任务的中止状态分为目标状态和非目标状态而构造。通过惩罚非目标状态，可以学习针对每个子任务的策略以鼓励在目标状态中止。由于“伪”奖励会扭曲子任务策略，因此这种以中止为导向的方法可能为 MDP 引入附加的亚优化性。

MAXQ 的重要特征是以分解的子任务完成价值与当下简单动作的期望奖励之和作为状态的价值。完成价值是采取下一个抽象动作后完成子任务的（折扣过的）期望累积奖励。

我们按照 [Dietterich, 2000] 的方法进行层次分解，并通过包含对特定子任务 m 的显式引用扩展 SMDP 概念。针对子任务 m 的公式 (9.3) 变为：

$$Q^\pi(m, s, a) = \sum_{s', N} T(m, s, a, s', N) [R(m, s, a, s', N) + \gamma^N Q^\pi(m, s', \pi(s'))] \quad (9.4)$$

针对子任务 m 的抽象动作 a 激活子任务 m_a ，完成子任务 m_a 的期望价值记作 $V^\pi(m_a, s)$ ，层次策略 π 是策略集合，其中每一个策略针对一个子任务。

完成函数（completion function） $C^\pi(m, s, a)$ 是在子任务 m 状态 s 下完成抽象动作 a 的期望折扣积累奖励，其折扣一直计算到 a 开始执行的时刻。

$$C^\pi(m, s, a) = \sum_{s', N} T(m, s, a, s', N) \gamma^N Q^\pi(m, s', \pi(s'))$$

子任务 m 的 Q 函数（公式 (9.4)）可以递归表示为完成 a 激活的子任务 m_a 的价值加上完成子任务 m 的价值。

$$Q^\pi(m, s, a) = V^\pi(m_a, s) + C^\pi(m, s, a) \quad (9.5)$$

完成子任务 m_a 的价值依赖于 a 是否是简单动作。如果 a 是简单动作，则 $V^\pi(m_a, s)$ 是在状态 s 执行动作 a 的期望奖励。

$$V^\pi(m_a, s) = \begin{cases} Q^\pi(m_a, s, \pi(s)) & , \text{是抽象动作} \\ \sum_{s'} T(s, a, s') R(s, a, s') & , \text{是简单动作} \end{cases} \quad (9.6)$$

如果从根节点子任务 m_a 到简单任务 m_k 的子任务序列为 m_0, m_1, \dots, m_k ，并且层次策略指定在子任务 m_i 的策略是 $\pi(s) = a_i$ ，那么公式 (9.5) 和公式 (9.6) 把价值函数 $Q^\pi(m_0, s, \pi(s))$ 分解为：

$$Q^\pi(m_0, s, \pi(s)) = V^\pi(m_k, s) + C^\pi(m_{k-1}, s, a_{k-1}) + \dots + C^\pi(m_1, s, a_1) + C^\pi(m_0, s, a_0)$$

为了得到在给定任务层次下的贪婪式最优解，为实现抽象动作 a 而对公式 (9.6) 的分解修改为选择最佳动作 a ，即 $V^*(m_a, s) = \max_a Q^*(m_a, s, a)$ 。引入最大值运算符 \max 意味着我们必须搜索任务层次中的所有路径才能找到最优解。算法 18 执行这样的深度优先方法，返回在状态 s 执行动作 a 的价值和最佳动作。

当任务层次的深度增加，以上穷举式搜索就会过于昂贵，此时控制复杂度的一种方法是限制深度 [ZHengst, 2004]。例如在规划国际旅行时，飞行和机场接送方法都需要考虑，但是就不必优化在启程那天应该从床的哪边走到浴室了。

```

Require:  $V(m, s)$  for primitive actions  $m$ , abstract actions  $A_m$ ,  $C(m, s, j) \forall j \in A_m$ 
1: if  $m$  is a primitive action then
2:   return  $\langle V(m, s), m \rangle$ 
3: else
4:   for each  $j \in A_m$  do
5:      $\langle V(j, s), a_j \rangle = \text{Evaluate}(j, s)$ 
6:    $j_{\text{greedy}} = \text{argmax}_j [V(j, s) + C(m, s, j)]$ 
7:   return  $\langle V(j_{\text{greedy}}, s), a_{j_{\text{greedy}}} \rangle$ 

```

算法 18 估计 (m, s) [Dietterich, 2000]

对于图 9.2 的四房间问题来说, 分解后学习器状态值有三个部分由任务的两个层次和简单动作决定。当学习器位于图 9.4 中实心椭圆所示的状态时, 最优价值函数就是离开房屋最短路径的代价。该代价通过累加采用向北简单动作、完成从西离开房间的低层子任务以及完成离开房屋的高层次任务这三部分的期望奖励而获得。

1. MAXQ 学习和执行

给定设计师制定的任务层次, 算法 19 是相应的在线 Q 学习算法 (见第 1 章), 能够为每个子任务 SMDP 找到完成函数。如果是简单动作, 该算法学习该动作在每个状态下的期望奖励 (第 3 行)。对于抽象动作来说, 该动作的完成函数将被更新 (第 12 行)。学习速率参数 α 逐渐收敛到 0。算法 19 是 MAXQ 算法的简化版。该算法的扩展版本加速了学习过程, 并对目标和非目标状态使用“伪奖励 (pseduo-reward)”进行区分 (参阅 [Dietterich, 2000])。算法 20 启动 MAXQ 过程进行任务层次上的学习和执行。

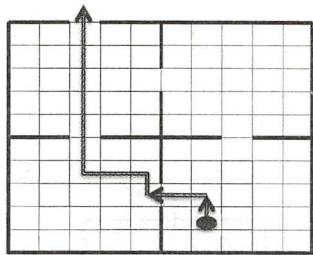


图 9.4 学习器根据最优策略求解图 9.1 中四房间问题时分解价值函数的任务模块。学习器用位于起始位置的实心椭圆表示

311

```

Require:  $V, C, A$ , learning rate  $\alpha$ 
1: if  $m$  is a primitive action then
2:   execute  $m$ , receive reward  $r$ , and observe the result state  $s'$ 
3:    $V(m, s) \leftarrow (1 - \alpha)V(m, s) + \alpha r$ 
4:   return 1
5: else
6:    $count = 0$ 
7:   while  $m$  has not terminated do
8:     choose an exploration action  $a$ 
9:      $N = MAXQ(a, s)$ 
10:    observe result state  $s'$ 
11:     $\langle V(j^{greedy}, s'), a_{j^{greedy}} \rangle \leftarrow Evaluate(m, s')$ 
12:     $C(m, s, a) \leftarrow (1 - \alpha)C(m, s, a) + \alpha \gamma^N V(j^{greedy}, s')$ 
13:     $count \leftarrow count + N$ 
14:     $s = s'$ 
15:   return  $count$ 

```

算法 19 MAXQ(m, s) [Dietterich 2000]

```

Require: root node  $m_0$ , starting state  $s_0, V, C$ 
1: initialise  $V(m, s)$  and  $C(m, s, a)$  arbitrarily
2: MAXQ( $m_0, s_0$ )

```

算法 20 MAXQ 的主函数

2. 应用到四房间任务的层次式强化学习方法

现在我们可以整合前述概念并演示怎样对图 9.1 的四房间任务进行学习和执行, 这里学习器可以从任何状态开始。图 9.5 任务层次的设计师意识到若干抽象状态的机会, 记住这里状态由二元组 (房间, 位置) 描述。

学习器可以从通向东、南、西、北的四个过道的任意一个离开房间, 因此我们需要为每一个选择学习一个单独的导航策略。但是, 由于房间都是相同的, 房间变量与房间内的导航无关。

312

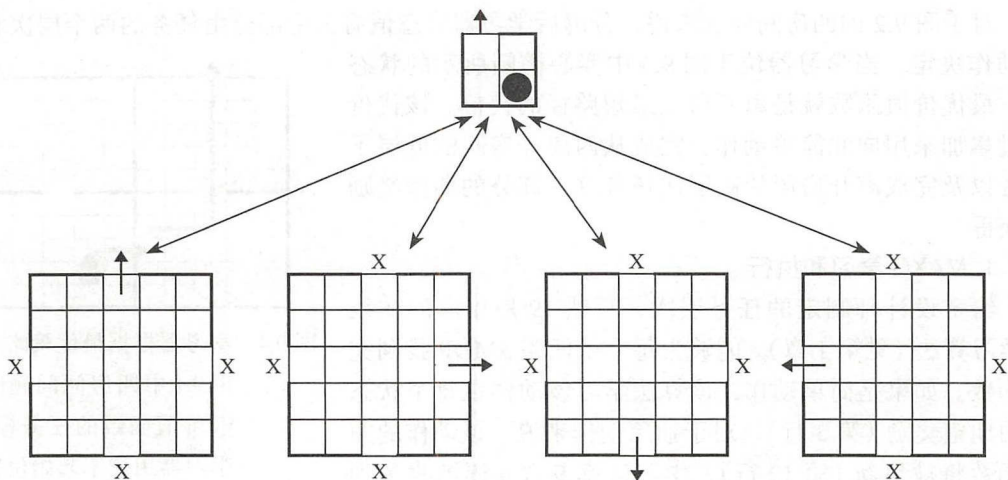


图 9.5 四房间任务的层次。子任务是离开房间的动作，可以在任意房间使用。父级根节点
子任务有 4 个状态对应于四个房间，“X”表示非终极目标状态

我们也注意到离开房间的抽象动作总是在一个状态结束。离开房间的抽象动作似乎把学习器“漏”出过道。因此，可以抽象掉房间内位置的状态，在根节点只需保留房间级状态，而且只需要考虑离开房间的抽象动作。这意味着我们不需要根级子任务的 100 个状态，而只需要对应于每个房间的 4 个状态，或者说我们只需要学习 16（即 4 个状态 \times 4 个动作）个完成函数，而不是 400 个。

为了使用算法 19 在任务层次上求解问题，主程序任意启动期望简单奖励函数 $V(\cdot, \cdot)$ 和完成函数 $C(\cdot, \cdot)$ ，并在根级子任务针对起始状态 s_0 调用 MAXQ 函数。MAXQ 使用类似 Q 学习的更新规则针对所有子任务去学习简单动作的期望奖励和完成函数。

当价值函数收敛时，设置 α 值为 0 并中止搜索过程。算法 18 的 MAXQ 按照递归最优策略执行，搜索离开四个房间的最短路径。图 9.4 画出了针对一个起始位置的一条路径和分解的价值函数。

9.4 学习结构

在大多数层次式强化学习应用中，任务层次的结构或者部分程序由设计师以背景知识形式提供。自动化问题分解看起来是更加困难的问题，用机器把设计师从这个任务中解放出来是大家乐意看到的。层次式结构需要以下知识：怎样最优化分解问题以及怎样引入平衡复杂度和优化程度的状态抽象。

读者可能熟悉分割过的棋盘问题，其中问题表征在求解中具有重要意义 [Gamow and Stern, 1958]。[Amarel, 1968] 论述了针对传教士和食人族问题的六种表征，展示了使得机器学习更加容易发现规则性的可行性以及利用这一能力产生新的表征方式。在增强式学习问题中，选择表征状态和动作的变量在形成分解问题的机会方面作用很大。

有些研究人员尝试从学习器-环境交互中学习层次结构。大多数方法寻找把问题分解为几乎独立的可用子问题的子目标或子任务。自动分解问题的方法包括寻找子目标瓶颈和地标状态以及寻找求解路线和区域的共性策略。



从底向上的学习

通过与环境交互自动学习复杂任务的分解要求按阶段的学习。如果程序员能够从顶向下指定动作的话,学习就像是从底向上演化的。

[Simon 1996]观察到:“如果具有稳定的中间形式则复杂系统可以从简单系统非常快速地演化。”[Clark and Thornton, 1997]提出了对复杂环境建模的一个很有说服力的观点,即需要从底向上求解简单问题作为中间表征。用他们的话说就是:

“...背后的技巧总是相同的,即最大化所能获得的表征的角色,从而最小化之后的搜索空间。”

这与[Stone 1998]提倡的原则类似,该原则把问题分解为多层抽象,从最底层向最高层的任务依次学习,其中底层的输出送往下一层作为输入。[Utgoff and Stracuzzi, 2002]指出从简单到复杂任务的进程中存在固有的可压缩性。他们建议一种基于积木块的方法,设计目的是减少知识结构的复制。学习器通过不断通过从底向上在现有概念上获取新概念移动其“感知边界(frontier of receptivity)”,从而不断扩展其知识。他们的结论是:

“通过把局部学习限制于简单任务并把学习结果用于后续学习可以成功地引导复杂结构的学习。”

有些方法可用于学习抽象动作和结构,包括搜索共性的动作序列和状态区域的策略[Thrun and Schwartz, 1995; McGovern, 2002]。其他方法寻找瓶颈或地标状态[Digney, 1998; McGovern, 2002; Menache et al, 2002]。[Simsek and Barto, 2004]使用一种较为新颖的方法发现子目标状态。有趣的是,[Moore et al, 1999]认为对某些探索任务而言,求解性能对路标位置并不敏感,一组(自动)随机产生的路标并不会比专门产生的路标为结果带来更大的扰动[Moore et al, 1999]。

314

HEXQ

现在我们来介绍一种自动学习层次结构的机器学习方法。HEXQ(Hierarchical Exit Q Function, 层次式退出的Q函数)是一系列受到MAXQ价值函数分解和由底向上策略启发而形成的算法。这些算法依赖于有限维度(分解过的)的底层状态描述。假定学习器并不知道潜在的MDP,目标是学习任务层次并找到最优策略。

HEXQ从底层的状态和简单动作开始构造层次。在无关于某些变量的状态空间中,HEXQ可以找到抽象动作。接下来,HEXQ利用之前找到的抽象动作的“漏斗”动作,递归地表述更高层次的、简化的SMDP问题。

1. 学习抽象动作和任务层次

通过探索投影状态空间的各个状态变量的转换和奖励函数,HEXQ寻找子空间。当子空间状态转换不影响其他变量并且转换和奖励函数与其他变量无关时,则可以划分至相同的状态分块。这是一种更严格形式的统计双仿同质(bisimulation homogeneity),见9.2.4节,此时整个环境在以不同于投影变量的形式下保持不变。这种联合的状态抽象从子空间中去除了无关的环境变量。

一旦一个状态转换违背以上条件,则需要产生一个由状态-动作对(s,a)表征的退出(exit)操作。如果退出无法由起始状态到达,那么分裂子空间并产生更多的退出操作。也就是说,产生退出操作是一种自动形成子目标的机制。

HEXQ分割投影至每个变量的状态空间。分割后的每一个状态块对应一组子任务,每个



子任务对应一个退出操作。子任务是一个 SMDP，其目标为通过执行退出操作而中止。处理中止条件由状态-动作对定义之外，HEXQ 的子任务类似于选项。中止的奖励不是子任务的一部分，而是在任务层次的更高级别考虑的。这实际上是对 MAXQ 价值函数进行了小的改造，从而统一了任务层次上的 Q 函数定义并简化了递归公式。

HEXQ 的早期版本在对状态变量根据其变化频率排序后学习单一的层次 [Hengst, 2002]。在较新的版本里面，状态变量可以被并行处理 [Hengst, 2008]。投影到变量的状态空间仍像以前一样进行拆分。父级状态变量由对不同块的子变量进行叉积 (cross-product) 产生。这种对因子化状态空间进行并行分解的方法产生顺序 (或多任务) 的一组动作，每个动作可以以顺序方式激活多个子任务。顺序动作形成了一个偏序 (partial-order) 的任务层次，该层次有可能实现更大程度的状态抽象 [Hengst, 2008]。

315

2. 四房间问题的 HEXQ 分解

对于四房间问题，HEXQ 提供了因子式的状态变量对 (房间，位置)。其求解始于为每一个变量产生一个模块，位置模块发现拥有四个出口的分块，这里出口是通过一个过道离开任意房间的转换。由于出口只用于房间变量可变的转换，因此可以被自动发现。位置模块会提出四个离开房间的子任务，每一个针对一个出口。子任务策略可以使用针对 SMDP 的标准 Q 学习离线策略。

房间模块学习分割由单一状态块构成的房间状态，这是由于在房间中执行简单动作可以改变位置变量的值。对每一个拥有四个出口的分块，分块的标志符就是房间状态，该标志符与来自位置模块的块标志符共同组成父级模块的一个新状态变量。

在这个例子中，房间模块并不为状态抽象增加任何价值，并且可以通过作为输入的一部分直接向父级模块传递从而加以消除。不管哪种方式，父级模块都可以表示抽象的房间状态，并且启动离开房间的抽象动作去实现目标。机器产生的任务层次类似于图 9.5 所示的层次。

有趣的是，如果四房间状态由坐标 (x,y) 定义，其中 x 和 y 的取值在 0 到 9 之间，如果不使用更具描述力的 (房间，位置)，则 HEXQ 就会产生表示四房间的更高层次的变量而不是由设计师定义。 x 和 y 变量被 HEXQ 分割成两块，分别表示房间内的空间。块标志符的叉积产生一个高层次变量，代表四个房间。HEXQ 产生东西和南北移动的顺序抽象动作离开房间。这种子空间的分解具有不同的意义，即对单独房间建模而不是使用一个通用的房间 [Hengst, 2008]。

3. HEXQ 摘要

316

通过假定一个潜在的有限状态分解的 MDP，HEXQ 从与环境的交互中自动建立任务层次。该方法使用变量消除和漏斗类型的状态抽象去构造简洁表征 (见 9.2.4 节)。由于为每一个出口学习一个子任务，HEXQ 的解是层次式最优的 (见 9.2.6 节)。HEXQ 可以使用层次贪婪式执行去尝试改进层次式最优解，同时 HEXQ 策略收敛于只有根子任务具有随机转换和奖励函数的任务层次的全局最优解。通过引入变量和顺序动作的并行分解，使得 HEXQ 能够产生设计师没有指定的抽象状态变量。在任意任务层次上定义的分解策略价值函数等同于该策略在原始 MDP 上执行的价值函数，这样的分解保证是“安全”的。

9.5 相关工作和当前研究

层次式强化学习 (HRL) 并非孤立的领域。结合本书其他章节介绍的题目，HRL 已经形



成了若干贡献,也提出了研究挑战。我们简述这些工作。

之前我们在处理 HRL 时采用了离散时间假设,但是 SMDP 是在更具一般性的连续时间时域扩展动作上定义的 [Puterman, 1994]。[Ghavamzadeh and Mahadevan, 2001] 针对第 1 章介绍的折扣奖励和平均奖励优化模型把 MAXQ 扩展到连续时间模型。[Ghavamzadeh and Mahadevan, 2003] 使用了层次策略的梯度方法去处理拥有连续状态和动作空间的强化学习问题。[Jong and Stone, 2009] 把函数近似和优化探索结合起来,从而允许 MAXQ 求解极大甚至无限的状态空间。

我们已经看到消除无关变量可以在不同环境之中隐式地迁移学习到的技能。层次分解在迁移学习中扮演着重要角色,这里迁移学习指从解决某个任务中获取的经验可以用来解决一个具有关联但又不同的问题。[Konidaris and Barto, 2007] 构造了能够区分学习器-空间和问题-空间的移植选项。以学习器为中心的学习器-空间表征与直立式 (deictic) 表示更加密切相关 [Agre and Chapman, 1987]。[Marthi et al, 2007] 使用样本轨迹去学习 MAXQ 风格的层次,其中目标函数衡量规划或增强式学习可以在未来环境中使用同样层次进行简化的程度。[Mehta et al, 2008a] 在不同 SMPD 之间迁移价值函数,并且发现迁移过程在层次设置下特别有效。[Mehta et al, 2008b] 通过在某个源增强式学习问题的成功轨迹上使用动态贝叶斯网络发现 MAXQ 任务层次。一个有趣的结论是迁移关于任务层次结构的知识比迁移价值函数更加重要。后者甚至可能阻碍新任务的学习,因为此时可能还需要通过学习丢掉迁移过来的策略。[Taylor and Stone, 2009] 综述了包括抽象动作、状态抽象和层次方法等各种增强式学习配置下的迁移学习技术。为了实现有效的知识迁移,两个环境需要“足够接近”。[Castro and Precup, 2010] 演示了使用一种双仿度量时迁移抽象动作比迁移简单动作更加有效。

317

针对类似于 MAXQ 的任务层次的状态抽象可能被折扣奖励最优模型所妨碍,这是由于采取抽象动作的奖励与完成抽象动作时间不再相关。[Hengst, 2007] 开发了一种在任务层次上分解折扣价值函数的方法,其中使用并行分解的多时间模型 [Precup and Sutton, 1997]。两个递归分解函数恢复状态抽象的机会,并允许在任务层次中具有连续子任务的问题可以被 HRL 求解。

增强式学习的很多文献包含一维动作。但是,在许多领域我们希望同时控制若干动作变量。例如在控制机器人队伍或者单个机器人拥有多个关节时,就会产生这种情况。这里的挑战是在任务层次上分解动作,特别是在存在抽象动作有可能交互的情况下更具挑战性。[Rohanimanesh and Mahadevan, 2001] 使用选项框架演示了怎样使用 SPMD 表述并行化动作。[Fitch et al, 2005] 用一个两出租车问题为例,揭示了用并行动作任务层次和状态抽象去降低包含并行动作的问题的复杂度。并行动作要求子任务中止的模式 [Rohanimanesh and Mahadevan 2005] 以及任务奖励属性。[Marthi et al, 2005] 通过使用多线程并行 ALisp 把部分程序的概念 (见 9.2.3 节) 扩展到并行动作。

当状态不是完全可观测或者观测过程存在噪声的时候,MDO 变成了部分可观察马尔可夫决策问题 (Partially Observable Markov Decision Problem, POMDP), 见第 12 章。[Wiering and Schmidhuber, 1997] 把 POMDP 分解为一系列简单子任务。[Hernandez and Mahadevan, 2000] 利用一个基于存储器的 SMDP 在长决策序列上传播奖励,从而解决部分可观察顺序决策任务。[Pineau and Thrun, 2002] 提出了在结构化 POMDP 上使用基于动作的分解把复杂问题切割为小的子问题层次从而进行规划的方法。[Theocharous and Kaelbling, 2004] 通过



扩展包括用多入口和出口状态标准子空间边界的研究工作,从隐式马尔可夫模型推导出一种层次式部分可观察决策问题(Hierarchical Partially Observable Markov Decision Problem, HPOMDP)。

318 目前结构学习技术仍在继续发展之中。原始的 HEXQ 分解使用简单的启发式测量决定分解时的状态空间顺序。[Jonsson and Barto, 2006] 提出一种基于贝叶斯网络因果图模型的方法——变量影响结构分析(Variable Influence Structure Analysis, VISA),把变量的互相影响与构造任务层次联系起来。与 HEXQ 不同,该算法整合了互相影响的变量,但是忽略底层活动。[Bakker and Schmidhuber, 2004] 提出的 HASSLE 算法通过学习抽象状态间的转换发现子目标。在此过程中,子目标抽象动作经过推广可以重用或定制,从而在状态空间的不同部分工作或者实现不同的目标。[Moerman, 2009] 使用函数近似对 HASSLE 进行了扩展。[Strehl et al, 2007] 使用一种因子式状态表征把学习贝叶斯网络结构作为强化学习的一部分。

[Mugan and Kuipers, 2009] 提出了一种通过学习连续环境下定量表征方式并找到到达定性状态的方法在连续环境中学习动作层次的算法。[Neumann et al, 2009] 提出了学习对一组构成连续时间下的抽象动作(选项)的运动模板进行参数化和排序。[Konidaris and Barto, 2009] 引入了一种为连续时间域强化学习发现技巧的方法,即构造走向结束任务奖励的技巧链。该方法被 [Konidaris et al, 2010] 进一步扩展,从而能够从样本求解轨迹中更快地构造技巧树。

[Osentoski and Mahadevan, 2010] 扩展了针对 HRL 的自动基函数构造。该方法建立在对 SMDP 状态空间中的求解轨迹相应的图结构进行层次谱分析的基础之上。[Mahadevan, 2010] 综述了近来在发现通用表征方面的最新进展。该综述论文包括了时域和同质状态抽象,并把后者推广到以基函数形式抽象地表征价值函数。

9.6 总结

层次式强化学习把问题分解为子任务的层次,其中高层次的父任务把低层次的子任务激活作为简单动作。问题分解可以包括多个级别的层次,其中部分或全部子任务均可以也是强化学习问题。当父任务被表述为强化学习问题时,通常是可以表述为半马尔可夫决策问题,这是因为其动作是持续一定时间的子任务。层次式分解的优势是在整个问题能够精简表征和存在可重用或独立的子任务时减少计算复杂度。通常在通过分解减少复杂度和分解后问题能够达到的优化程度方面存在折衷。

参考文献

- Agre, P.E., Chapman, D.: Pengi: an implementation of a theory of activity. In: Proceedings of the Sixth National Conference on Artificial Intelligence, AAAI 1987, vol. 1, pp. 268–272. AAAI Press (1987)
- Amarel, S.: On representations of problems of reasoning about actions. In: Michie, D. (ed.) Machine Intelligence, vol. 3, pp. 131–171. Edinburgh at the University Press, Edinburgh (1968)
- Andre, D., Russell, S.J.: Programmable reinforcement learning agents. In: Leen, T.K., Dietterich, T.G., Tresp, V. (eds.) NIPS, pp. 1019–1025. MIT Press (2000)
- Andre, D., Russell, S.J.: State abstraction for programmable reinforcement learning agents. In: Dechter, R., Kearns, M., Sutton, R.S. (eds.) Proceedings of the Eighteenth National Conference on Artificial Intelligence, pp. 119–125. AAAI Press (2002)



- Ashby, R.: Design for a Brain: The Origin of Adaptive Behaviour. Chapman & Hall, London (1952)
- Ashby, R.: Introduction to Cybernetics. Chapman & Hall, London (1956)
- Bakker, B., Schmidhuber, J.: Hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization. In: Proceedings of the 8-th Conference on Intelligent Autonomous Systems, IAS-8, pp. 438–445 (2004)
- Barto, A.G., Mahadevan, S.: Recent advances in hierarchical reinforcement learning. Special Issue on Reinforcement Learning, Discrete Event Systems Journal 13, 41–77 (2003)
- Bellman, R.: Adaptive Control Processes: A Guided Tour. Princeton University Press, Princeton (1961)
- Boutillier, C., Dearden, R., Goldszmidt, M.: Exploiting structure in policy construction. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, vol. 2, pp. 1104–1111. Morgan Kaufmann Publishers Inc., San Francisco (1995)
- Boutillier, C., Reiter, R., Soutchanski, M., Thrun, S.: Decision-theoretic, high-level agent programming in the situation calculus. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, pp. 355–362. AAAI Press (2000)
- Brooks, R.A.: Elephants don't play chess. Robotics and Autonomous Systems 6, 3–15 (1990)
- Castro, P.S., Precup, D.: Using bisimulation for policy transfer in mdps. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010, vol. 1, pp. 1399–1400. International Foundation for Autonomous Agents and Multiagent Systems, Richland (2010)
- Clark, A., Thornton, C.: Trading spaces: Computation, representation, and the limits of uninformed learning. Behavioral and Brain Sciences 20(1), 57–66 (1997)
- Dayan, P., Hinton, G.E.: Feudal reinforcement learning. In: Advances in Neural Information Processing Systems (NIPS), vol. 5 (1992)
- Dean, T., Givan, R.: Model minimization in Markov decision processes. In: AAAI/IAAI, pp. 106–111 (1997)
- Dean, T., Lin, S.H.: Decomposition techniques for planning in stochastic domains. Tech. Rep. CS-95-10, Department of Computer Science Brown University (1995)
- Dietterich, T.G.: Hierarchical reinforcement learning with the MAXQ value function decomposition. Journal of Artificial Intelligence Research 13, 227–303 (2000)
- Digney, B.L.: Learning hierarchical control structures for multiple tasks and changing environments. From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behaviour SAB (1998)
- Ferrein, A., Lakemeyer, G.: Logic-based robot control in highly dynamic domains. Robot Auton. Syst. 56(11), 980–991 (2008)
- Fitch, R., Hengst, B., Šuc, D., Calbert, G., Scholz, J.: Structural Abstraction Experiments in Reinforcement Learning. In: Zhang, S., Jarvis, R.A. (eds.) AI 2005. LNCS (LNAI), vol. 3809, pp. 164–175. Springer, Heidelberg (2005)
- Forestier, J., Varaiya, P.: Multilayer control of large Markov chains. IEEE Transactions Automatic Control 23, 298–304 (1978)
- Gamow, G., Stern, M.: Puzzle-math. Viking Press (1958)
- Ghavamzadeh, M., Mahadevan, S.: Continuous-time hierarchical reinforcement learning. In: Proc. 18th International Conf. on Machine Learning, pp. 186–193. Morgan Kaufmann, San Francisco (2001)
- Ghavamzadeh, M., Mahadevan, S.: Hierarchical policy gradient algorithms. In: Marine Environments, pp. 226–233. AAAI Press (2003)
- Hauskrecht, M., Meuleau, N., Kaelbling, L.P., Dean, T., Boutillier, C.: Hierarchical solution of Markov decision processes using macro-actions. In: Fourteenth Annual Conference on Uncertainty in Artificial Intelligence, pp. 220–229 (1998)
- Hengst, B.: Discovering hierarchy in reinforcement learning with HEXQ. In: Sammut, C., Hoffmann, A. (eds.) Proceedings of the Nineteenth International Conference on Machine Learning, pp. 243–250. Morgan Kaufmann (2002)
- Hengst, B.: Model Approximation for HEXQ Hierarchical Reinforcement Learning. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS



- (LNAI), vol. 3201, pp. 144–155. Springer, Heidelberg (2004)
- Hengst, B.: Safe State Abstraction and Reusable Continuing Subtasks in Hierarchical Reinforcement Learning. In: Orgun, M.A., Thornton, J. (eds.) AI 2007. LNCS (LNAI), vol. 4830, pp. 58–67. Springer, Heidelberg (2007)
- Hengst, B.: Partial Order Hierarchical Reinforcement Learning. In: Wobcke, W., Zhang, M. (eds.) AI 2008. LNCS (LNAI), vol. 5360, pp. 138–149. Springer, Heidelberg (2008)
- Hernandez, N., Mahadevan, S.: Hierarchical memory-based reinforcement learning. In: Fifteenth International Conference on Neural Information Processing Systems, Denver (2000)
- Hutter, M.: Universal algorithmic intelligence: A mathematical top→down approach. In: Artificial General Intelligence, pp. 227–290. Springer, Berlin (2007)
- Jong, N.K., Stone, P.: Compositional models for reinforcement learning. In: The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (2009)
- Jonsson, A., Barto, A.G.: Causal graph based decomposition of factored mdps. *Journal of Machine Learning* 7, 2259–2301 (2006)
- Kaelbling, L.P.: Hierarchical learning in stochastic domains: Preliminary results. In: Proceedings of the Tenth International Conference Machine Learning, pp. 167–173. Morgan Kaufmann, San Mateo (1993)
- Konidaris, G., Barto, A.G.: Building portable options: skill transfer in reinforcement learning. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 895–900. Morgan Kaufmann Publishers Inc., San Francisco (2007)
- Konidaris, G., Barto, A.G.: Skill discovery in continuous reinforcement learning domains using skill chaining. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) *Advances in Neural Information Processing Systems*, vol. 22, pp. 1015–1023 (2009)
- Konidaris, G., Kuindersma, S., Barto, A.G., Grun, R.: Constructing skill trees for reinforcement learning agents from demonstration trajectories. In: *Advances in Neural Information Processing Systems NIPS*, vol. 23 (2010)
- Korf, R.E.: *Learning to Solve Problems by Searching for Macro-Operators*. Pitman Publishing Inc., Boston (1985)
- Levesque, H., Reiter, R., Lespérance, Y., Lin, F., Scherl, R.: Golog: A logic programming language for dynamic domains. *Journal of Logic Programming* 31, 59–84 (1997)
- Mahadevan, S.: Representation discovery in sequential decision making. In: 24th Conference on Artificial Intelligence (AAAI), Atlanta, July 11–15 (2010)
- Marthi, B., Russell, S., Latham, D., Guestrin, C.: Concurrent hierarchical reinforcement learning. In: *Proc. IJCAI 2005 Edinburgh, Scotland* (2005)
- Marthi, B., Kaelbling, L., Lozano-Perez, T.: Learning hierarchical structure in policies. In: *NIPS 2007 Workshop on Hierarchical Organization of Behavior* (2007)
- McGovern, A.: Autonomous Discovery of Abstractions Through Interaction with an Environment. In: Koenig, S., Holte, R.C. (eds.) *SARA 2002. LNCS (LNAI)*, vol. 2371, pp. 338–339. Springer, Heidelberg (2002)
- Mehta, N., Natarajan, S., Tadepalli, P., Fern, A.: Transfer in variable-reward hierarchical reinforcement learning. *Mach. Learn.* 73, 289–312 (2008a), doi:10.1007/s10994-008-5061-y
- Mehta, N., Ray, S., Tadepalli, P., Dietterich, T.: Automatic discovery and transfer of maxq hierarchies. In: *Proceedings of the 25th International Conference on Machine Learning, ICML 2008*, pp. 648–655. ACM, New York (2008b)
- Menache, I., Mannor, S., Shimkin, N.: Q-Cut - Dynamic Discovery of Sub-goals in Reinforcement Learning. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) *ECML 2002. LNCS (LNAI)*, vol. 2430, pp. 295–305. Springer, Heidelberg (2002)
- Moerman, W.: Hierarchical reinforcement learning: Assignment of behaviours to subpolicies by self-organization. PhD thesis, Cognitive Artificial Intelligence, Utrecht University (2009)
- Moore, A., Baird, L., Kaelbling, L.P.: Multi-value-functions: Efficient automatic action hierarchies for multiple goal mdps. In: *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1316–1323. Morgan Kaufmann, San Francisco (1999)



- Mugan, J., Kuipers, B.: Autonomously learning an action hierarchy using a learned qualitative state representation. In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pp. 1175–1180. Morgan Kaufmann Publishers Inc., San Francisco (2009)
- Neumann, G., Maass, W., Peters, J.: Learning complex motions by sequencing simpler motion templates. In: *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009*, pp. 753–760. ACM, New York (2009)
- Nilsson, N.J.: Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research* 1, 139–158 (1994)
- Osentoski, S., Mahadevan, S.: Basis function construction for hierarchical reinforcement learning. In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010*, vol. 1, pp. 747–754. International Foundation for Autonomous Agents and Multiagent Systems, Richland (2010)
- Parr, R., Russell, S.J.: Reinforcement learning with hierarchies of machines. In: *NIPS (1997)*
- Parr, R.E.: Hierarchical control and learning for Markov decision processes. PhD thesis, University of California at Berkeley (1998)
- Pineau, J., Thrun, S.: An integrated approach to hierarchy and abstraction for pomdps. *CMU Technical Report: CMU-RI-TR-02-21* (2002)
- Polya, G.: *How to Solve It: A New Aspect of Mathematical Model*. Princeton University Press (1945)
- Precup, D., Sutton, R.S.: Multi-time models for temporally abstract planning. In: *Advances in Neural Information Processing Systems*, vol. 10, pp. 1050–1056. MIT Press (1997)
- Puterman, M.L.: *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York (1994)
- Ravindran, B., Barto, A.G.: SMDP homomorphisms: An algebraic approach to abstraction in semi Markov decision processes. In: *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, IJCAI 2003* (2003)
- Rohanimesh, K., Mahadevan, S.: Decision-theoretic planning with concurrent temporally extended actions. In: *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pp. 472–479. Morgan Kaufmann Publishers Inc., San Francisco (2001)
- Rohanimesh, K., Mahadevan, S.: Coarticulation: an approach for generating concurrent plans in Markov decision processes. In: *ICML 2005: Proceedings of the 22nd international conference on Machine learning*, pp. 720–727. ACM Press, New York (2005)
- Russell, S., Norvig, P.: *Artificial Intelligence: A Modern Approach*. Prentice Hall, Upper Saddle River (1995)
- Ryan, M.R.K.: Hierarchical Decision Making. In: *Handbook of Learning and Approximate Dynamic Programming*. IEEE Press Series on Computational Intelligence. Wiley-IEEE Press (2004)
- Reid, M.D., Ryan, M.: Using ILP to Improve Planning in Hierarchical Reinforcement Learning. In: Cussens, J., Frisch, A.M. (eds.) *ILP 2000*. LNCS (LNAI), vol. 1866, pp. 174–190. Springer, Heidelberg (2000)
- Si, J., Barto, A.G., Powell, W.B., Wunsch, D.: *Handbook of Learning and Approximate Dynamic Programming*. IEEE Press Series on Computational Intelligence. Wiley-IEEE Press (2004)
- Simon, H.A.: *The Sciences of the Artificial*, 3rd edn. MIT Press, Cambridge (1996)
- Şimşek, O., Barto, A.G.: Using relative novelty to identify useful temporal abstractions in reinforcement learning. In: *Proceedings of the Twenty-First International Conference on Machine Learning, ICML 2004* (2004)
- Singh, S.: Reinforcement learning with a hierarchy of abstract models. In: *Proceedings of the Tenth National Conference on Artificial Intelligence* (1992)
- Stone, P.: Layered learning in multi-agent systems. PhD, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA (1998)
- Strehl, A.L., Diuk, C., Littman, M.L.: Efficient structure learning in factored-state mdps. In: *Proceedings of the 22nd National Conference on Artificial Intelligence*, vol. 1, pp. 645–650. AAAI Press (2007)
- Sutton, R.S., Precup, D., Singh, S.P.: Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112(1-2), 181–211



- (1999)
- Taylor, M.E., Stone, P.: Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research* 10(1), 1633–1685 (2009)
- Theocharous, G., Kaelbling, L.P.: Approximate planning in POMDPS with macro-actions. In: *Advances in Neural Information Processing Systems 16 (NIPS-2003)* (2004) (to appear)
- Thrun, S., Schwartz, A.: Finding structure in reinforcement learning. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) *Advances in Neural Information Processing Systems (NIPS)*, vol. 7. MIT Press, Cambridge (1995)
- Utgoff, P.E., Straczuzi, D.J.: Many-layered learning. In: *Neural Computation*. MIT Press Journals (2002)
- Watkins CJCH, Learning from delayed rewards. PhD thesis, King's College (1989)
- Wiering, M., Schmidhuber, J.: HQ-learning. *Adaptive Behavior* 6, 219–246 (1997)



针对强化学习的演化计算

Shimon Whiteson

摘要

通过模仿自然选择过程而优化问题求解的演化计算算法，是一种发现高性能强化学习策略的有效工具。由于能够自然寻找问题表征方式、处理连续动作空间并解决部分可观察性，演化强化学习技术在实际问题上表现出色，经常能够达到比时序差分方法更好的性能。本章综述演化计算在强化学习中的应用，总结演化神经网络的拓扑和权重、使用时序差分方法的混合方法、针对多学习器环境的协同演化方法、生成式和发展式系统以及在线演化式强化学习。

10.1 简介

针对演化计算的算法有时也称作遗传算法 (genetic algorithms) [Holland, 1975; Goldberg, 1989]，是通过模仿自然选择过程发现对某一问题具有高度适应性的解的优化方法。一般来说，相应问题会假定一个适应函数 $f: C \rightarrow \mathbb{R}$ ，把所有潜在解的集合 C 映射到实数值的适应度上。优化方法的目标是找到最适应的解 $c^* = \operatorname{argmax}_c f(c)$ 。在某些例子中，适应函数可以是随机的，其中 $f(c)$ 可以看作随机变量，此时 $c^* = \operatorname{argmax}_c E[f(c)]$ 。

演化方法通过重复地选择和复制一个潜在解的“人口”集合搜索 c^* 。初始人口集合一般随机选择，之后集合中的成员通过函数 f 进行评估，最优成员被选中作为下一代人口的基础。新一代人口由复制产生，其中复制策略可以是交叉（即整合两个不同的解的不同部分）和突变（即一个解的某些参数值随机改变）。该过程重复多次，直至找到具有足够适应性的解或者计算资源耗尽。

325

该算法的变体很多，例如多目标方法 [Deb, 2001; Coello et al, 2007]、多样化算法 [Holland, 1975; Goldberg and Richardson, 1987; Mahfoud, 1995; Potter and De Jong, 1995; Darwen and Yao, 1996] 和分布式方法 [Larranaga and Lozano, 2002; Hansen et al, 2003; Rubinstein and Kroese, 2004]。然而，基本的方法是非常通用的，而且可以应用到任何能够定义 f 的优化问题上。

这些优化问题也包括强化学习任务 [Moriarty et al, 1999]。在这种情况下， C 对应于可能策略的集合，例如从 S 到 A 的映射，而 $f(c)$ 就是使用该策略在一系列蒙特卡罗随机实验中的累积奖励。换言之，在使用演化算法的强化学习中，算法直接搜索策略空间，寻找能够最大化期望累积奖励的解。

类似于其他许多策略搜索方法，演化算法只考虑整个策略的价值，而不会像时序差分方法那样为具体的状态-动作对构造价值估计。这种整体式的本质也经常受到诟病。例如，Sutton 和 Barto 指出：



“演化方法没有利用这样一个事实，即所搜索的策略是一个从状态到动作的函数；该方法没有考虑个体在生命周期中经过的状态及其选择的动作。虽然在某些情况下这些信息有可能导致误导（例如状态感知错误时），但是更多时候可以使得搜索过程更加有效率 [Sutton and Barto, 1998, p. 9]。”

这些事实使得演化方法在理论上具有劣势。例如在某些环境下，动态规划方法可以保证在多项式状态和动作个数复杂度内找到最优策略 [Littman et al, 1995]。对照来看，演化方法在最坏情况下必须遍历指数级个数的候选策略才能找到最优解。实际结果也表明演化方法比时序差分方法需要更多的训练周期才能找到优秀策略，特别是在高度随机任务中，只有经过大量蒙特卡罗仿真才能找到每个候选策略适应度的可靠估计 [Runarsson and Lucas, 2005; Lucas and Runarsson, 2006; Lucas and Togelius, 2007; Whiteson et al, 2010b]。

尽管有这些局限性，演化计算仍然是解决强化学习问题的一种广受欢迎的工具，并且在多个领域获得成功，有些情况下还能够大幅度超越时序差分方法的性能 [Whitley et al, 1993; Moriarty and Miikkulainen, 1996; Stanley and Miikkulainen, 2002; Gomez et al, 2008; Whiteson et al; 2010b]。主要原因有以下三个方面。

326 首先，演化方法可以处理部分可观察性。演化方法并不利用学习器访问的候选状态之间的关系，这样的特点在学习器不知道自身状态时是很有利的。由于时序差分方法显式地依赖于马尔可夫特性，其价值估计在马尔可夫特性不满足时将会发散，从而导致对贪婪式策略来说灾难性的后果。对比而言，演化方法不依赖于马尔可夫特性，总会选择其能找到的最好策略。虽然严重的部分可观察性为相应策略的性能给出了上界，但是在给定策略空间中的优化可以正常进行 [Moriarty et al, 1999]。此外，使用存储器减少部分可观察的表征方法，例如递归神经网络，可以使用演化方法很自然地进行优化 [Gomez and Miikkulainen, 1999; Stanley and Miikkulainen, 2002; Gomez and Schmidhuber, 2005a,b]。

其次，演化方法能够简化寻找解的适当表征形式，由于策略只需要为每个状态指定一个动作，而不是每个状态-动作对的价值，因此更易于表征，同时，也可以并发演化适当的策略表征（见 10.3 节和 10.4.2 节）。不仅如此，由于不需要对给定潜在解进行学习更新，可以使用更加精细的表征方式，比如生成和发展系统（Generative and Developmental Systems, GDS）采用的方法，见 10.6 节。

第三，演化方法提供了更容易解决拥有大的或者连续动作空间的问题的方法。由于需要在动作空间中的每个状态迭代从而找到最好的动作，许多时序差分方法并不适合这样的任务。对比而言，演化方法只需要直接将状态映射至动作的进化策略。当然，actor-critic 方法 [Doya, 2000; Peters and Schaal, 2008] 和其他技术 [Gaskett et al, 1999; Millán et al, 2002; van Hasselt and Wiering, 2007] 也可以用来使得时序差分方法适合于连续动作空间。然而，演化方法提供了一种简单而有效的路径来解决以上困难。

当然，以上这些问题也并非演化方法所独有，原则上对其他策略方法也可以成立。但是，演化方法被证明是一种特别受欢迎的搜索策略空间的方法，也因此出现了大量针对强化学习的算法和成果。不仅如此，当以分布式技术为例的现代方法越来越偏离最初的遗传算法时，它们与自然选择过程也越来越不相似了。因此，演化方法与其他策略搜索技术的边界越来越模糊、越来越不重要。

本章介绍并综述针对强化学习的演化方法。该领域之庞大，使得回顾所有重要发展和技术变得不再现实。为了清晰和简洁，本章侧重于神经演化（neuroevolution）[Yao, 1999]，

其中演化方法用于神经网络的进化 [Haykin, 1994], 从而表征策略。虽然演化强化学习绝不只限于神经网络表征方式, 但神经演化是目前最常用的技术。不仅如此, 由于神经网络一般说来也是一种广受欢迎和广泛研究的表征方法, 所以也适合作为本章的焦点。

327

本章组织如下。作为导论, 10.2 节描述了一种简单的神经演化强化学习算法。10.3 节讨论基于拓扑和基于权重的人工神经网络 (Topology- and Weight-Evolving Artificial Neural Networks, TWEANN), 包括很受欢迎的 NEAT 方法, 该方法能够自动发现自身的内在表征。10.4 节介绍混合方法, 包括演化函数近似和学习分类系统, 这些方法集成了时序差分和演化。10.5 节讨论了协同演化 (coevolution), 允许多个互相竞争或协作的策略共同演化。10.6 节论述以 HyperNEAT 为例的生成和发展系统, HyperNEAT 依赖于间接编码, 即更复杂的表征方式, 其中学习器策略必须从演化的参数值中构造或生长。10.7 节讲述在线方法, 该方法尽量最大化演化过程中的奖励, 而不是仅仅快速发现优秀的策略。

10.2 神经演化

神经网络 [Haykin, 1994] 是异常通用的表示复杂函数的方式。由于其简洁性, 神经网络是一种广泛使用的强化学习策略表征方式, 不仅用于演化计算, 也可用于其他策略搜索算法和时序差分方法。本节介绍神经演化强化学习的基础, 其中演化算法用于优化神经网络策略。

图 10.1 演示了神经演化算法的基本步骤 [Yao, 1999]。在每一代上, 人口集中的神经网络需要针对任务进行评估。其次, 性能最佳的网络将通过基于名次的选择、轮盘赌式选择或竞赛式选择等方法被选中 [Goldberg and Deb, 1991]。选中的网络通过交叉和突变进行复制 [Sywerda, 1989; De Jong and Spears, 1991], 从而产生新一代人口并再次重复以上过程。

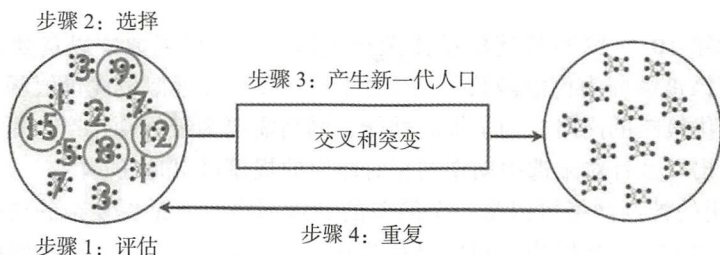


图 10.1 神经演化基本步骤

328

在强化学习背景下, 网络中的每一个输入节点通常对应于一个状态特征, 这样所有输入值共同描述了学习器的状态。表示动作的方法有多种。如果动作也可以被特征描述, 就像连续动作空间中常见的那样, 则每个输出节点可以对应于一个动作特征。在这种情况下, 输出值共同描述了针对特定状态的动作选择。当动作数量较小并且是离散类型时, 每个动作可以有一个单独的网络, 就像在价值函数近似中常见的情况。针对某一状态的策略动作则对应于产生最大输出的网络。

本章中, 我们侧重于一种称为动作 - 选择网络的方法。像以前一样, 假定简单动作集合较小并且是离散类型。然而, 只有一个表示策略的网络。该网络为每个动作设置一个输出节点。策略针对一个给定状态的动作相当于为该状态生成最大的输出。

算法 21 是一个简单神经演化方法的高层次描述, 该方法针对一个阶段式 (episodic) 强化学习问题演化动作 - 选择网络。算法起始于产生一个随机网络群体 (第 4 行)。在每一代

群体上, 算法重复遍历现有个体 (第 6 ~ 7 行)。在某阶段的每一个步骤上, 学习器选择最大激活程度的输出所对应的动作 (第 10 ~ 12 行)。注意 s' 和 a' 是当前的状态和动作, 而 s 和 a 是之前的状态和动作。神经演化算法维护其网络在评估过程中的累积奖励的记录 (第 13 行)。每一代在 e 个阶段后结束, 在结束点上网络的评均适应度由 $N.fitness/N.episodes$ 计算。在随机域上, e 通常必须大于 $|P|$ 从而确保准确的适应度估计。神经演化通过反复调用 BREED-NET 函数 (第 18 行) 产生新一代网络群体, 该函数从最适应的父辈网络中生成新的网络。

```

1: // S: set of all states, A: set of all actions, p: population size
2: // g: number of generations, e: episodes per generation
3:
4:  $P \leftarrow \text{INIT-POPULATION}(S, A, p)$  // create new population P with random networks
5: for  $i \leftarrow 1$  to  $g$  do
6:   for  $j \leftarrow 1$  to  $e$  do
7:      $N, s, s' \leftarrow P[j \% p]$ , null, INIT-STATE(S) // select next network
8:     repeat
9:        $Q \leftarrow \text{EVAL-NET}(N, s')$  // evaluate selected network on current state
10:       $a' \leftarrow \text{argmax}_i Q[i]$  // select action with highest activation
11:       $s, a \leftarrow s', a'$ 
12:       $r, s' \leftarrow \text{TAKE-ACTION}(a')$  // take action and transition to new state
13:       $N.fitness \leftarrow N.fitness + r$  // update total reward accrued by N
14:    until TERMINAL-STATE?(s)
15:     $N.episodes \leftarrow N.episodes + 1$  // update total number of episodes for N
16:     $P' \leftarrow$  new array of size  $p$  // new array will store next generation
17:    for  $j \leftarrow 1$  to  $p$  do
18:       $P'[j] \leftarrow \text{BREED-NET}(P)$  // make a new network based on fit parents in P
19:     $P \leftarrow P'$ 

```

算法 21 神经演化算法 (S, A, p, g, e)

注意, 尽管第 10 ~ 12 行描述的动作选择类似于从价值函数中以贪婪方式进行动作选择, 该网络不应该被解释为价值函数^①。演化并不寻找最佳近似最优价值函数的网络, 而是寻找代表性能最优策略的网络。为了提高性能, 网络需要为最优动作产生更多的输出。不同于价值函数, 输出以及针对未选中动作的相对输出的规模可以是任意的。

算法 21 使用的神经网络可以是一个简单的前馈网络或者是更复杂的递归网络。递归神经网络可以包含环路 (例如输出节点的结果送往输入节点)。于是, 这些网络可能包含内部状态。在强化学习背景下, 内部状态可以记录学习器观测历史的各个方面, 从而帮助其处理部分观测性的情况 [Wieland, 1991; Gomez and Miikkulainen, 1999; Moriarty et al, 1999; Stanley and Miikkulainen, 2002; Igel, 2003; Gomez and Schmidhuber, 2005a, b]]。

算法 21 使用了传统的遗传算法对神经网络的权重进行优化, 实际上存在很多变体。例如也可以使用分布估计算法 (Estimation of Distribution Algorithm, EDA) [Larranaga and Lozano, 2002; Hansen et al, 2003; Rubinstein and Kroese, 2004]。EDA 算法也称为概率建模遗传算法 (Probabilistic Model Building Genetic Algorithm, PMBGA), 并不显式地维护候选解的种群集合, 而是维护解的分布, 在每一代上, 候选解从该分布中采样形成并进行评估。密度估计方法使用选中的子集更新概率分布, 其中的密度估计可以使用非监督学习技术根据一部分样本来近似概率分布。

协方差矩阵适应演化策略 (Covariance Matrix Adaptation Evolution Strategy, CMA-ES)

① 这一点不能用于 10.4 节讨论的混合方法。

[Hansen et al, 2003] 是最常用、最有效的 EDA 算法之一, 该算法是一种变量度量 EDA, 其中概率分布是一个协方差矩阵随时间改变的多变量高斯分布。当被用来优化神经网络时, 相应的 CMA-NeuroES 方法对大范围强化学习任务均非常有效 [Igel, 2003; Heidrich-Meisner and Igel, 2008, 2009a,b,c]。

10.3 TWEANN

在其最简单的形式中, 算法 21 仅演化固定表征形式的神经网络。这样的背景下, 某一特定演化阶段上的所有网络均有相同的拓扑, 也就是说隐变量节点的个数和连接节点的边集合都是固定的。种群中网络的差异只在于边上的权重, 而权重是通过演化优化的。使用固定的表征当然并非神经演化所独有。实际上, 尽管存在自动发现价值函数量化表征的方法 [Mahadevan and Maggioni, 2007; Parr et al, 2007], 时序差分方法也常常为函数近似采用固定表征。 [330]

然而, 依赖于固定表征的方式存在重大局限性。主要的原因是这样就需要算法使用者事先就正确地指定良好的表征方法。显然, 选择过于简化的表征注定会导致恶劣的性能, 这时描述高质量的解就不可能了。但是, 选择过于复杂的表征也同样有害。虽然能够描述良好的求解结果, 找到这些结果反而变得不可行。由于网络中每一个权重均相当于搜索空间的一维, 拥有过多边的表征可能导致无法计算的搜索问题。

在多数情况下, 使用者无法猜测正确的表征形式。技术在使用者拥有极佳的领域知识的情况下, 从专业知识中推导正确的表征形式仍然是不可能的。通常, 找到良好表征只能是一个试错的过程。不过, 重复进行演化直至发现适当的表征将会极大增加计算成本。不仅如此, 对于在线任务 (见 10.7 节) 来说, 在目标环境尝试各种策略也增加了实际成本。

由于以上原因, 许多研究者考查怎样自动化发现良好表征方式的方法 [Dasgupta and McGregor, 1992; Radcliffe, 1993; Gruau, 1994; Stanley and Miikkulainen, 2002]。演化方法非常适合上述挑战, 这是因为其采用一种直接搜索策略的方法解决强化学习问题。特别是, 由于神经演化已经直接搜索网络权重空间, 所以也可以同时搜索网络拓扑空间, 这种方法有时叫作拓扑和权重演化人工神经网络 (Topology- and Weight-Evolving Artificial Neural Network, TWEANN)。

结构遗传算法 (structured Genetic Algorithm, sGA) 也许是最早和最简单的 TWEANN [Dasgupta and McGregor, 1992], 该算法使用两部分的表征描述一个网络。第一部分以二值矩阵形式表示网络的连接关系, 其中行和列对应于网络中的节点, 每个数值指出相应节点对之间是否存在一条边。第二部分表示网络中边的权重。原则上, 通过演化连接和权重矩阵, sGA 可以自动发现适当的网络拓扑。但是, sGA 存在若干局限性。下面的几个小节里, 我们讨论这些局限性, 从而揭示 TWEANN 面临的主要挑战。 [331]

10.3.1 挑战

成功开发 TWEANN 需要应对三方面的挑战, 第一个挑战就是“竞争合约” (coming convention) 问题。多数情况下会出现许多具有相似适应度的策略, 例如, 许多任务会包含对称性并由此造成等价解。这会给演化带来困难, 其原因在于由于演化方法依赖于交叉操作产生新一代种群。当两个表示不同策略的网络整合时, 其结果可能是破坏性的, 即导致一个对任何一个父辈网络来说都不适合的策略。

虽然竞争合约问题存在于任意使用交叉操作的演化方法, 但该问题对 TWEANN 尤为突出。两个父辈网络可能不仅实现不同的策略, 而且需要不同的表征。因此, TWEANN 需要一种整合具有不同拓扑的网络的机制, 该机制应该能够尽量减少灾难性交叉的可能性。显然, 由于在演化二值矩阵时没有对交叉操作考虑表征方式的兼容性, sGA 不能够应对这种挑战。实际上, “竞争合约” 问题也是早期 TWEANN 的主要障碍, 直到研究者们开发出完全不依赖于交叉操作的方法, 从而简单地回避了这一困难 [Radcliffe, 1993]。

第二个挑战是需要把拓扑上的创新保持足够长时间, 从而优化相关权重。通常, 当引入了新的拓扑结构时 (例如增加一个新的隐藏节点或边), 其结果往往对适应度有不利影响, 尽管这可能对长远结果的更好策略是必需的。其原因就在于相应新结构的权重尚没有优化。

例如我们考虑由 sGA 算法演化的网络中一条尚未被激活的边, 也就是其在二值矩阵中相应数值为 0。由于还没有在网络中显示作用, 这条边的权重还没有经历任何选择压力。如果演化过程突然激活这条边, 由于其权重尚未优化, 因此对于适应度的效果似乎是损害性的。因此, 如果拓扑上的创新没有被刻意保护的话, 通常会从种群中消除, 从而导致结构上的搜索处于停滞。

幸运的是, 在演化运算中保护拓扑创新已经受到广泛研究。物种形成和细分 (speciation and niching) 方法通常通过隔离不同个体或惩罚过于相似的个体保证种群的多样性 [Holland, 1975; Goldberg and Richard-son, 1987; Mahfoud, 1995; Potter and De Jong, 1995; Darwen and Yao, 1996]。但是, 使用这些方法需要能够量化个体差异的距离度量。为 TWEANN 设计这种度量是相当困难的, 原因在于不清楚怎样比较具有不同拓扑的网络。

第三个挑战是怎样演化最简化的解。前面已经说过, TWEANN 的一个核心出发点就是希望避免过于复杂的拓扑。然而, 如果像许多具体 TWEANN 算法那样, 演化起始于随机选择的网络, 其中某些的拓扑可能已经过于复杂。这样一来, 至少一部分演化搜索被分配在其实没有必要的高维空间上。一种可能是通过在适应度中增加尺度惩罚从而显式地奖励更简化的解 [Zhang and Muhlenbein, 1993]。不过目前尚没有主流的方法在缺乏任务需要的拓扑复杂度的先验知识时确定尺度惩罚方式。

10.3.2 NEAT

神经演化增强拓扑 (NeuroEvolution of Augmenting Topologies, NEAT) 可能是最流行的 TWEANN [Stanley and Miikkulainen, 2002]。本节简述 NEAT, 并解释该方法怎样应对前述的挑战。

像 10.3.2 节描述的那样, NEAT 经常用于演化动作选择。实际上, NEAT 遵循算法 21 所示的算法框架, 与传统神经演化方法的不同只在于怎样初始化种群和产生新一代种群。

为了能够表征变化拓扑的网络, NEAT 使用一种灵活的遗传编码。每个网络由一系列边基因描述, 其中每个基因刻画两个节点基因中的一条边。边基因指定输入节点、输出节点和边的权重。在变化过程中, 网络可以通过特定变异运算符增加新的节点和边基因从而引入新的结构 (见图 10.2)。

NEAT 通过记录每个基因历史来源的创新数 (innovation number) 来避免灾难性的交叉。突变产生一个新的基因时, 该基因会接受一个独特的创新数。因此, 创新数可以看作在演化过程中产生的所有基因的编年记录。

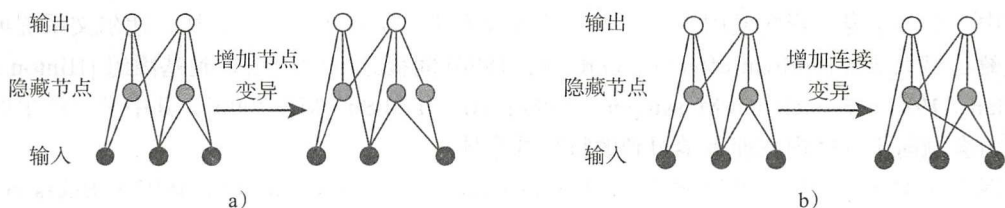


图 10.2 NEAT 的结构变异运算符。a) 分裂已有节点从而引入新节点。b) 在两个已知节点之间引入一条新的边

在交叉过程中，用创新数确定两个父辈基因的关系。依赖于创新数是否在其父辈创新数范围之内还是之外，不匹配的基因要么不相交、要么过剩。在交叉时，拥有相同创新数的两个来自不同父辈的基因可以被排在一起，不匹配的基因则继承于更具有适应度的一个父辈。这种方法允许 NEAT 最小化灾难性交叉的机会，同时又不需要使用昂贵的拓扑分析。由于对

333

应不同拓扑的基因在演化过程中保持兼容，NEAT 从本质上避免了竞争合约问题。

创新数也允许一种保护拓扑创新性的简单方法。特别是，NEAT 使用创新数在拓扑创新的基础上标识种群。两个网络编码的距离 δ 就是过剩基因个数 (E)、不相交基因个数 (D) 以及匹配基因的平均权重差 (W) 的线性组合：

$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \cdot \overline{W}$$

其中的系数 c_1 、 c_2 和 c_3 调整以上三个因素的重要程度，而 N 用于将更大基因组中的基因个数归一化为基因组大小。距离超过兼容阈值 δ_i 的网络被认为是不同物种。显式适应度共享 (explicit fitness sharing) 方法使用这样的方法保护创新物种，即同一物种的网络必须共享其细分区域的适应度 [Goldberg, 1989]。

为了鼓励演化出最简化的解，NEAT 起始于简单网络的均匀种群，这些简单网络没有隐藏节点，其输入直接连接到输出。新的结构以增量方式通过产生隐藏节点和边的变异操作引入。由于仅有产生性能优势的结构编译能够在选择压力下生存，该方法实际上偏好简化的解。

NEAT 已经在许多困难的强化学习问题上获得了可观的成功，这些问题包括非马尔可夫双平衡杆 [Stanley and Miikku lainen, 2002]、游戏 [Stanley and Miikkulainen, 2004b] 和机器人控制 [tanley and Miikkulainen, 2004a; Taylor et al, 2006; Whiteson et al, 2010b]。但是，[Kohl and Miikkulainen 2008, 2009] 指出，当最优动作在状态上不连续变化时，NEAT 的性能会比较差。他们演示了怎样通过引入具有本地接受域的神经元和约束拓扑搜索到级联结构来缓解以上问题。

10.4 混合方法

许多研究者分析了怎样结合演化和监督式或半监督式学习方法。在这样的系统中，演化个体不是在其适应度评估过程中保持不变，而是在生命期中从交互环境中不断学习。

许多混合方法的工作侧重于分析由演化和学习交互作用而产生的动力学。例如，若干工作 [Whitley et al, 1994; Yamasaki and Sekiguchi, 2000; Pereira and Costa, 2001; Whiteson and Stone, 2006a] 使用混合方法比较拉马克和达尔文系统。在拉马克系统中，基因表现型的学习效果在复制之前被拷贝回基因组，从而允许后代继承。在与生物学更为密切的达尔文

334

系统中,学习本身不影响基因组。正如其他混合方法的研究所表明的那样,达尔文系统可以通过鲍德温效应(Baldwin effect) [Baldwin, 1896] 间接迁移学习结果到基因组 [Hinton and Nowlan, 1987; French and Messinger, 1994; Arita and Suzuki, 2000], 其中学习产生的选择压力会倾向于选择内在拥有学习到的特征的个体。

混合方法也用于改进监督学习任务的性能 [Gruau and Whitley, 1993; Boers et al, 1995; Giraud-Carrier, 2000; Schmidhuber et al, 2005, 2007]。但是, 因为缺少必需的标签数据, 这些方法不能直接用于强化学习问题。

然而, 许多针对强化学习的混合方法已经逐渐发展起来。为了回避标签缺失的问题, 研究者使用了非监督学习技术 [Stanley et al, 2003], 例如训练个体使其与父辈类似 [McQuesten and Miikkulainen, 1997]、培训个体预测状态转换 [Nolfi et al, 1994] 或者训练个体进行自我教学 [Nolfi and Parisi, 1997]。不过, 也许最自然的强化学习混合方法是结合演化和时序差分方法 [Ackley and Littman, 1991; Wilson, 1995; Downing, 2001; Whiteson and Stone, 2006a]。本节中, 我们将综述两种这样的混合方法: 演化函数近似 (evolutionary function approximation) 和学习分类系统 XCS。

10.4.1 演化函数近似

演化函数近似是一种综合演化和时序差分的方法, 能够自动选择函数近似表征从而运行高效的个体学习 [Whiteson and Stone, 2006a]。主要思想是这样的, 如果演化过程以进化价值函数而不是动作选择为导向, 则价值函数可以在每次评估适应度的时候被时序差分方法更新。这么一来, 系统可以演化其函数近似使之更加适合时序差分方法。这种直觉与生物学的组合已经用于很多计算系统 [Hinton and Nowlan, 1987; Ackley and Littman, 1991; Boers et al, 1995; French and Messinger, 1994; Gruau and Whitley, 1993; Nolfi et al, 1994], 能够生成有效的强化学习算法。本节我们简要介绍一种通过神经网络函数近似结合 NEAT 和 Q 学习算法而来的演化函数近似技术——NEAT+Q。

335 为了使得 NEAT 优化价值函数而不是动作选择, 其实只需要重新解释其输出的价值。神经网络动作选择的结构 (即每个输入对应一个状态特征和每个输出对应一个动作) 已经与 Q 学习函数近似相同。因此, 如果 NEAT 演化的网络权重是在其适应度评估过程中由 Q 学习和后向传播更新的话, 则也可以有效地演化价值函数 (而不是动作选择)。所以, 输出不再是任意的价值, 而是代表相应状态-动作对的长期折扣价值, 而且不仅可以用于选择最适当的动作, 并且可以更新其他状态-动作对的估计。

算法 22 列出了 HEAT+Q 的内层循环, 其中第 9~13 行取代了算法 21 原来的内容。每当学习器执行动作时, 该网络针对 Q 学习的目标进行后向传播 (第 7 行), 此时进行 ϵ 贪婪选择 (第 4~5 行)。图 10.3 刻画了整个算法: 从一个种群中选择网络进行评估, 此时产生的 Q 值用于选择动作。从环境产生的反馈用来执行时序差分更新和评估网络适应度 (即从选项中获得的整体奖励)。

类似于其他混合方法, NEAT+Q 整合了时序差分和演化方法的优点。特别是, 该方法巩固了 NEAT 算法发现有效表征的能力, 并利用该能力辅助神经网络价值函数近似。不同于传统的神经网络价值函数近似 (像是把所有鸡蛋放在一个篮子里, 这些方法依靠手工设计的单一网络表征价值函数), NEAT+Q 探索这些网络的空间从而增加了找到高性能表征的机会。因此, 该方法在某些任务上能够大幅度超越单独的时序差分和神经演化方法 [Whiteson and

Stone, 2006a]。

```
1: //  $\alpha$ : learning rate,  $\gamma$ : discount factor,  $\lambda$ : eligibility decay rate,  $\epsilon$ : exploration rate
2:
3:  $Q \leftarrow \text{EVAL-NET}(N, s')$  // compute value estimates for current state
4: with-prob( $\epsilon$ )  $a' \leftarrow \text{RANDOM}(A)$  // select random exploratory action
5: else  $a' \leftarrow \text{argmax}_k Q[k]$  // or select greedy action
6: if  $s \neq \text{null}$  then
7:    $\text{BACKPROP}(N, s, a, (r + \gamma \max_k Q[k]), \alpha, \gamma, \lambda)$  // adjust weights
8:  $s, a \leftarrow s', a'$ 
9:  $r, s' \leftarrow \text{TAKE-ACTION}(a')$  // take action and transition to new state
10:  $N.\text{fitness} \leftarrow N.\text{fitness} + r$  // update total reward accrued by  $N$ 
```

算法 22 NEAT+Q 的内循环

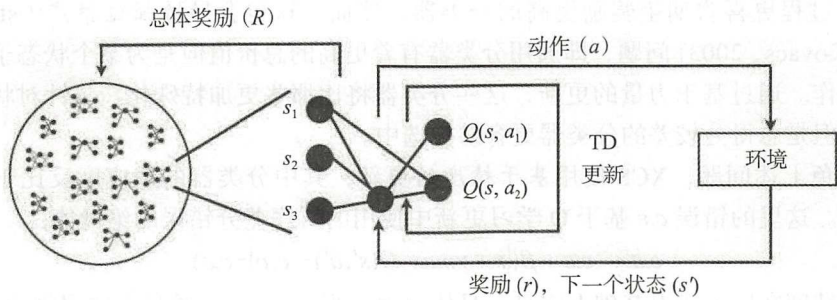


图 10.3 NEAT+Q 算法

10.4.2 XCS

使用学习分类器系统 (Learning Classifier System, LCS) 可以构造一种不同的混合方法 [Holland, 1975 ; Holland and Reitman, 1977 ; Bull and Kovacs, 2005 ; Butz, 2006 ; Drugowitsch, 2008]。LCS 是一个使用称为分类器的规则来表征解的演化系统。诸多分类、回归和优化问题可以被该系统解决。这些问题也包括强化学习问题，其中一系列分类器刻画了策略。

每个分类器包含一个反映其使用状态的条件。条件可以用多种方法指定 (例如模糊条件 [Bonarini, 2000] 或者神经网络条件 [Bull and O'Hara, 2002])，但是最简单的形式是用于二值状态的三元字母表 0、1 和 #，其中 # 表示“无关”。例如，具有条件 01#10 的分类器应用于状态 01010 和 01110。分类器同样包含动作和预测，前者用于界定学习器在分类器适用的状态上可以执行的动作，后者估计相应动作 - 价值函数。

在匹兹堡风格的分类器系统 [De Jong et al, 1993] 中，演化种群的每个个体由描述完整策略的一组规则组成。这些方法可以看作标准的演化强化学习，其中规则集取代神经网络表征策略的一种方法。对比而言，密歇根风格的分类器中，每个个体由单一规则组成。因此，整个种群表示一个单一的、演化的策略。在本节后续部分，我们简述最流行的密歇根风格分类器之一——XCS [Wilson, 1995, 2001 ; Butz et al, 2008]。由于其预测更新基于 Q 学习，XCS 可以看作时序差分和演化强化学习算法的混合。

在 XCS 中，每个分类器的预测向该分类器适用的状态 - 动作对的 Q 值贡献一个估计值。 $Q(s,a)$ 是所有适用于状态 s 并使用动作 a 的分类器预测值的加权平均。分类器的适应度决定其在平均值的权重 (适应度在本节后续部分定义)，换言之：

$$Q(s, a) = \frac{\sum_{c \in M(s, a)} c.f \cdot c.p}{\sum_{c \in M(s, a)} c.f}$$

[337] 其中 $M(s, a)$ 是匹配 s 和 a 的所有分类器的集合, $c.f$ 和 $c.p$ 分别是分类器 c 的适应度和预测值。

每当学习器到达状态 s 、执行动作 a 、接收到奖励 r 并转换到状态 s' 时, 下面的更新规则就被应用到每一个 $c \in M(s, a)$:

$$c.p \leftarrow c.p + \beta[r + \gamma \max_{a'} Q(s', a') - c.p] \frac{c.f}{\sum_{c' \in M(s, a)} c'.f}$$

其中 β 是学习速率的参数。由于上式决定分类器 c 对 Q 值的贡献, 除了更新的尺度与 c 的相对适应度成比例以外, 从本质上是 Q 学习更新。

许多 LCS 系统使用基于力量的更新, 其中每个分类器的适应度建立在其预测值的基础上, 即演化过程更喜欢期望奖励更高的分类器。然而, 这也会导致强过推广 (strong over-general) [Kovacs, 2003] 问题, 即通用分类器有着更高的总价值但是为某个状态子集选择了亚优化的动作。通过基于力量的更新, 这些分类器将比那些更加特殊化、只针对状态子集选择更好动作但是总得分较差的分类器更容易被选中。

为了避免上述问题, XCS 使用基于精度的更新, 其中分类器的适应度反比于预测值中的错误估计。这里的错误 $c.\varepsilon$ 基于 Q 学习更新中使用的时序差分错误的绝对值:

$$c.\varepsilon \leftarrow c.\varepsilon + \beta[|r + \gamma \max_{a'} Q(s', a') - c.p| - c.\varepsilon]$$

然后分类器精度在以上错误基础上定义。具体来说, 当 $c.\varepsilon > \varepsilon_0$ (ε_0 是最小错误阈值) 时, c 的精度定义为:

$$c.\kappa = \alpha(c.\varepsilon / \varepsilon_0)^{-\eta}$$

其中 α 和 η 是精度的参数。当 $c.\varepsilon \leq \varepsilon_0$ 时, $c.\kappa = 1$ 。

但是, 适应度不通过上述精度计算, 而是通过集合相对精度, 即精度除以所有匹配分类器的精度之和。这样我们就有下述适应度更新方法:

$$c.f \leftarrow c.f + \beta\left(\frac{c.\kappa}{\sum_{c' \in M(s, a)} c'.\kappa} - c.f\right)$$

在每个时间步长上, 所有匹配分类器的预测、错误和适应度均会更新。由于 XCS 是稳态演化方法, 因此没有代的概念, 而是种群通过周期性地选择和复制一部分适应性较好的分类器从而增量式改变, 其中一些较弱的分类器将被取代, 其余则保持不变。当选择发生时, 只有匹配当前状态和动作的分类器才被考虑。新的分类器从入选分类器经过交叉和突变产生, 而若干从所有分类器中选择的弱分类器将会被删除从而省出空间。

[338] 由于 Q 学习更新, 分类器的精度倾向于随时间而改进。由于稳态演化, 最准确的分类器进行选择繁衍。一般性的规则容易由于推广到过多状态而导致较大错误, 从而损失精度。这样看来, XCS 只会演化高度专用的规则。然而, 更一般性的规则更容易得到匹配。由于只有匹配的分类器才能繁衍, XCS 能够平衡特殊规则和一般性规则的压力。这样一来, XCS 可以学习更为全面、尽量一般性而又准确的分类器集合, 从而近似最优 Q 函数。

尽管尚没有 XCS 在 MDP 问题的收敛性证明, 该方法在许多实际问题上的表现极其有效。例如在迷宫任务上, XCS 证明能够很有技巧地自动发现状态特征忽视的问题 [Butz et al, 2005], 并且解决百万个状态级别的问题 [Butz and Lanzi, 2009]。该方法也能够有效处理复杂的神经运动控制 [Butz and Herbort, 2008; Butz et al, 2009] 和自主机器人问题 [Dorigo and Colombetti, 1998]。

10.5 协同演化

协同演化是来自演化生物学的概念,指多个共同演化的个体之间的交互。换言之,在单个个体的适应度函数依赖于其他演化个体时发生协同演化。在本质上,协同演化在以猎豹和被其捕猎的瞪羚为例的不同种群交互或以竞争求偶为例的单一种群交互的情况下发生。不仅如此,协同演化可以是合作性的,例如人和消化系统中的菌群,也可以是竞争性的,例如捕食动物和猎物。上述形式的协同演化均已在演化强化学习背景下获得了考察和应用,本节对此进行综述。

10.5.1 合作式协同演化

最显著的合作式协同优化应用是多学习器系统的协作 [Panait and Luke, 2005],其中演化的学习器团队在强化学习背景下协调各自的动作解决顺序决策问题。原则上,这种问题可以被没有协同的单一技术路线解决,即演化一个种群,其中每个个体都为其他学习器指定策略。但是,由于搜索空间随学习器的个数以指数级速度增长,这样的方法很快就会变得过于昂贵。

使用协同演化方法的主要动机是该方法有助于解决上述困难 [Wiegand et al, 2001; Jansen and Wiegand, 2004; Panait et al, 2006]。正如 [Gomez et al, 2008]所指出的那样,“许多问题可以分解为弱耦合的低维度子空间,这些子空间可以被不同物种半独立地搜索”。一般来说,确定低维度子空间需要大量领域知识。但是,在多学习器问题上,通常由学习器分割问题就已经足够,也就是说,为每个学习器演化一个种群从而使得演化过程不至于过分昂贵。使用这种方法,每次通常随机选择种群的一个成员去组成一个被评估的团队。总体奖励形成对每个参与学习器的适应度估计,其中的奖励一般需要多次评估。

[339]

尽管上述方法经常比单一演化的性能更好,并在捕食者-猎物 [Yong and Miikkulainen, 2007] 和机器人控制 [Cai and Peng, 2002] 等领域成功应用,在学习器数量较大时仍会遇到困难。主要问题在于单一学习器对总体奖励的贡献变得微不足道。这样一来,单个学习器的适应度更多地依赖于使用其策略评估哪些队友。但是,仍然可以为学习器构造特殊的适应度函数,从而对上述问题不那么敏感 [Agogino and Tumer, 2008]。主要的想法是使用差分函数 [Wolpert and Tumer, 2002] 比较该个体参与时的团队总体奖励和该个体不参与或被取代时的奖励。虽然该方法需要获取环境模型并且增加适应度函数的计算成本(这样在两种场景下的奖励均可评估),但能够显著提升合作式协同演化的性能。

协同演化也可以用于同时演化单个学习器的多个模块,而不是多个学习器。例如,在仿真机器人足球任务中,可以使用领域知识将任务分解为若干模块,每个模块代表诸如向球跑或跑出空挡准备接球 [Whiteson et al, 2005] 的重要技能。然后,为每个此类模块协同演化神经网络,并且共同组成完整策略。在仿真机器人足球任务中,协同演化大幅度超越了单一演化技术。

协同演化也能够用于单学习器设置中以协助神经演化。不同于演化多个网络,其中每个网络针对团队中的一个成员或策略中的一个模块,而是让神经元协同演化,并且这些神经元共同组成一个描述学习器策略的单个网络 [Potter and De Jong, 1995, 2000]。一般说来,这种网络拥有固定拓扑,其中包含一层隐藏层,每个神经元对应一个隐藏节点以及输入和输出边上的权重。类似于把多学习器任务按照学习器切分经常可以导致更简单的问题一样,同样

可以按照神经元拆分神经演化网络。正如 [Moriarty and Miikkulainen, 1997] 所指出的, “神经元级演化利用单个神经元构成神经网络的基本单元这一先验知识”。

340

这样的例子是共生适应性神经演化 (Symbiotic Adaptive NeuroEvolution, SANE) [Moriarty and Miikkulainen, 1996, 1997], 其中演化在两个层次上同时发生。底层演化一个神经元种群, 每个神经元的适应度就是其在使用适应度评估过程中所参与的神经网络的平均性能。高层演化一个蓝图 (blueprint) 种群, 其中每个蓝图由指向底层神经元的一组指针构成。在诸如机器人控制和平衡杆等强化学习任务中, SANE 表现出比时序差分、单纯神经演化和没有蓝图的神经元级演化更好的性能。

强制子种群 (Enforced SubPopulation, ESP) 方法 [Gomez and Miikkulainen, 1999] 去除了蓝图种群, 把神经元分成子种群, 每个子种群对应于一个隐藏节点。随机选中每个种群的一个成员以形成针对适应度评估的网络。这种方法鼓励子种群去扮演各种特化角色, 提高了即使没有蓝图也能够产生有效网络的可能性。ESP 在部分可观察任务上表现尤为出色, 能够解决双平衡杆的非马尔可夫版本。此外, ESP 使用蓝图的层次化变体 H-ESP, 在需要成千上万时间步长的深度记忆 (deep memory) POMDP 上表现出色 [Gomez and Schmidhuber, 2005a]。ESP 甚至被用到无鳍研究火箭的控制系统 [Gomez and Miikkulainen, 2003]。

协同突触神经演化 (Cooperative Synapse NeuroEvolution, CoSyNE) [Gomez et al, 2006, 2008] 进一步拓展可分离子种群的思想。CoSyNE 不为每一个神经元 (包含多个权重) 安排子种群, 而是为每条边 (只有一个权重) 设置一个子种群 (见图 10.4)。这样一来, 找到完整网络的问题就被拆分为一系列原子单元, 这些单元的解进而被协同演化。在平衡杆问题的几个版本中, CoSyNE 表现出能够超越时序差分方法和诸如 SANE、ESP 和 NEAT 等策略搜索方法的性能 [Gomez et al, 2006, 2008]。然而在较近的研究工作中, CMA-NeuroES (见 10.2 节) 被证明表现更佳 [Heidrich-Meisner and Igel, 2009b]。

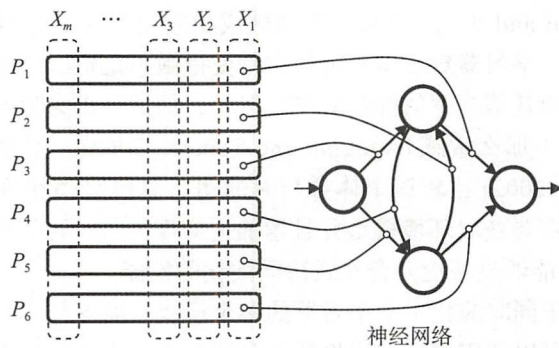


图 10.4 使用了 6 个子种群 (各包含 m 个权重) 的 CoSyNE 算法。索引值为 i 的所有权重组成一个基因原型 X_i 。每个权重来自一个不同子种群, 描述神经网络中的一条边。该图授权自 [Gomez et al, 2008]

341

10.5.2 竞争式协同演化

协同演化在竞争环境下同样是一种强大的工具。最常见的应用是游戏, 其中协同演化用于同时演化表现出色的玩家及对他们进行评估的对手。这里的希望是产生一种“军备竞赛”, 其中演化学习器彼此施加竞争压力, 使得演化向增加有效策略的方法发展 [Dawkins

and Krebs, 1979]。

也许竞争式协同演化最简单的例子是 [Pollack and Blair, 1998] 在西洋双陆棋 (backgammon) 游戏上的工作。他们的方法依赖于一种简单的优化技术 (基本上就是一个种群数量规模为 2 的演化方法), 其中神经网络与自身的编译版本对抗, 胜者将会生存。该技术的表现如此之好, 以至于 Pollack 和 Blair 认为 Tesauro 在 TD-Gammon 上的成功 [Tesauro, 1994] 应该归功于西洋双陆棋的本质, 而不是时序差分方法[⊖]。

使用较大种群时, 竞争式协同演化方法也在西洋跳棋上取得了成果。Blondie24 程序使用 minimax 算法进行对弈 [Von Neumann, 1928], 依靠神经演化去发现期盼位置的有效评估 [Chellapilla and Fogel, 2001]。在适应度评估过程中, 当前种群的成员互相对弈。尽管尽量少使用人类领域技能, Blondie24 能够演化到可以和人类专家竞争的水平。

竞争式协同演化在与 TWEANN 共同使用时也可以非常有效。在固定拓扑的神经演化中, 如果额外的改进需要扩展表征, 军备竞赛过程可以被缩短。由于 TWEANN 能够自动扩展其表征, 协同演化可以产生连续复杂化 [Stanley and Miikkulainen, 2004a]。

上述方法只演化单一种群。但是, 类似于合作式协同演化, 通过把个体分隔成分离的种群可以在某些情况下提高性能。在宿主 / 寄生者模型 [Hillis, 1990] 中, 一个种群演化出宿主和寄生者, 宿主在其相对于寄生者的鲁棒性基础上评估, 例如在西洋跳棋中可以击败多少寄生者。相比之下, 寄生者在其独特性基础上评估, 例如可以打败多少别的寄生者不能打败的宿主。这样的适应度函数可以被竞争适应度共享方法实现 [Rosin and Belew, 1997]。

在帕累托协同演化 (Pareto coevolution) 中, 该问题被处理为多目标问题, 每个对手作为一个目标 [Ficici and Pollack, 2000, 2001]。目标是找到帕累托最优解, 即无法在损害其他目标的前提下优化某一目标的解。许多方法使用这种思路, 通过维护对手的帕累托档案来评估演化的解 [De Jong, 2004; Monroy et al, 2006; De Jong, 2007; Popovici et al, 2010]。

[342]

10.6 生成和发展系统

所有上述演化强化学习方法均依赖于表征策略的直接编码。在这样的表征中, 演化过程优化一个基因型 (例如一个指定神经网络权重的向量), 该基因型可以被简单地转换为表现型 (例如神经网络自身)。虽然这种思路的简洁性具有吸引力, 但其性能扩展性存在局限。由于基因型总是和表现型规模相同, 演化解决许多实际任务的复杂策略往往需要搜索高维空间。

生成和发展系统 [Gruau, 1994; Hornby and Pollack, 2002; Stanley and Miikkulainen, 2003; Stanley et al, 2009] 是演化计算的一个子领域, 侧重于演化间接编码。在这样的表征上, 表现型通过一个由基因型制定的过程“生长”而来。在许多情况下, 好的策略具有简化的规则性, 例如对称性和重复性, 从而允许基因型的数量远小于其产生的表现型的数量。于是, 搜索基因型空间更为可行, 从而能够处理更大的强化学习问题。

不仅如此, 间接编码经常提供了一种利用“任务几何”(即状态特征的空间关系)的自然方式。由于这种几何不能被表征所捕捉, 所以无法被多数直接编码利用。例如我们可以考虑一个神经网络, 其中每个输入描述棋盘上一个方块的当前状态。由于将这些输入看作无序集合, 方块之间的距离就没能在这种表征中捕捉。因此, 利用方块间关系的结构需要被演化出

⊖ 但是 Tesauro 反对这种说法, 指出与 Pollack 和 Blair 的方法之间性能差距极大, 类似于一般人类玩家和世界冠军的差距 [Tesauro, 1998]。

来，这当然使得问题复杂化。对比而言，间接编码可以描述这样一个网络，其中处理方块的结构是一个方块位置的函数，这样一来附近的方块可以类似处理。

类似于其他演化方法，使用间接编码的系统也是受到生物学相似性的启发而来。例如，人类身体拥有数以万亿计的细胞，但是却由包含数以十万计基因的基因组发展而来。因此，许多间接编码在自然发展的模型基础上产生。一个例子是 L 系统 [Lindenmayer, 1968]，一种描述复杂递归结构的形式化语法，已经用于演化自主机器人的形态和控制系统并且大幅度超越了直接编码的性能 [Hornby and Pollack, 2002]。类似的，细胞化编码 (cellular encoding) 演化图语法，从而产生由简单子网络构成的模块化网络 [Gruau and Whitley, 1993; Gruau, 1994]。该方法允许演化过程为同一子网络产生多个副本，从而利用解结构中的规则性构造一个完整的网络。

近期的工作中，HyperNEAT 方法 [Stanley et al, 2009] 使用间接编码扩展了 NEAT。该方法建立在组合模式生成网络 (Compositional Pattern Producing Network, CPPN) 的基础之上。CPPN 是描述复杂模式的神经网络。例如，二维图片可以被 CPPN 描述，其中输入对应于图片中的 (x, y) 位置，输出对应于相应位置的颜色。该图片就可以通过从 CPPN 中检索每个 (x, y) 位置并根据输出设置该处颜色而产生。这样的 CPPN 可以由 NEAT 演化而成，从而形成一个发展式系统，其中 CPPN 是基因型、图片是表现型。

在 HyperNEAT 中，CPPN 可用来描述神经网络而不是图片本身。这样，基因型和表现型都表现为神经网络。如图 10.5 所示，表现型网络的节点置于一个基底即网格之上，每个节点均有一个位置。CPPN 以两个位置 (而非一个) 作为输入，其输出指定连接两个节点之间的权重。像以前一样，这些 CPPN 由 NEAT 根据相应表现型网络的适应度 (例如其作为一种策略在强化学习任务中的性能) 进行演化。CPPN 可以解释为描述四维超空间中的模式，由此产生了 HyperNEAT 这一称谓。由于这种发展式技术使得定义网络更容易，从而利用复杂任务的对称性和规则性，已证明 HyperNEAT 是一种有效的强化学习技术，在西洋跳棋 [Gauci and Stanley, 2008, 2010]、机器人足球仿真 [Verbancsics and Stanley, 2010] 和多学习器系统 [D'Ambrosio et al, 2010] 等领域均取得了成功应用。

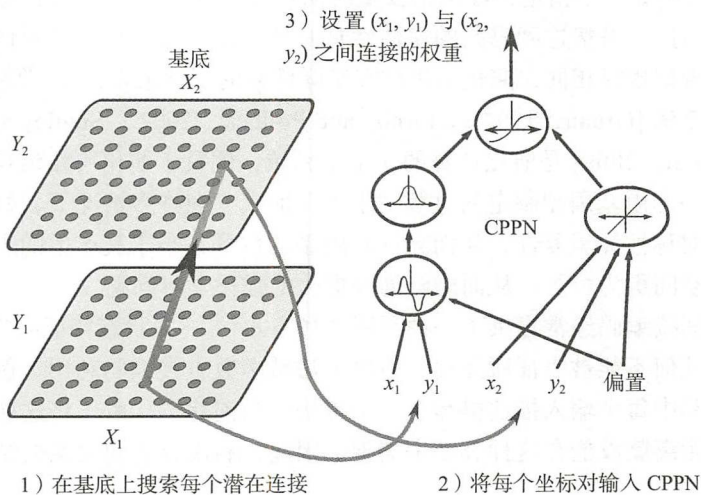


图 10.5 HyperNEAT 算法。该图授权自 [Gauci and Stanley, 2010]

10.7 在线方法

虽然演化方法在许多极具挑战性的强化学习问题上表现出色,其实用价值仍然基本限于离线场景,即学习器不是在真实环境中而是在诸如仿真器的“安全”环境中学习。换言之,问题表述通常包含适应度函数,正如由演化算法处理的其他优化问题一样,该函数只需要对计算资源进行评估。

在离线背景之下,学习器的唯一目标是尽快学习优秀策略。在学习过程中获得多少奖励是不相关的,这是因为这些奖励只是假设的而不对应于真实世界的成本。如果学习器尝试了灾难性策略,也只有计算时间的损失而已。

有效率的离线学习固然是重要的目标,但其应用性却要打个折扣。在许多情况下,由于任务动态不可知(例如机器人探索未知环境或者象棋选手与新对手对弈),因此没有相应的仿真器,其他情况下,任务动态经历过于复杂,难以精确仿真,典型例子是大型计算机网络上的用户动作以及机器人传感器和执行器的噪声。

因此,许多研究人员把在线学习作为强化学习的一个基本挑战。在线学习场景之中,学习器必须最大化其在学习过程中获得的奖励,因为这些奖励就对应于真实世界的成本。例如,如果在线学习的机器人尝试一个使其开下悬崖的策略,则其损失(即负奖励)就不再是假设性的,而是对应于维修或更换机器人的成本。

演化方法已经找到成功的在线应用 [Steels, 1994; Nordin and Banzhaf, 1997; Schroder et al, 2001], 特别是在演化机器人方面 [Meyer et al, 1998; Floreano and Urzelai, 2001; Floreano and Mondada, 2002; Pratihari, 2003; Kernbach et al, 2009; Zufferey et al, 2010], 其中一些研究还考察了为在线场景定制相应方法 [Floreano and Urzelai, 2001; Whiteson and Stone, 2006a,b; Priesterjahn et al, 2008; Tan et al, 2008; Cardamone et al, 2009, 2010]。

然而,在大多数应用中,研究人员通常报告只使用优化问题常用的离线度量时的性能,例如找到具备阈值性能的策略之前需要对适应度进行评估的次数或者在一定评估次数之后找到的最优策略的性能。因此,决定在线环境下怎样针对强化学习使用演化方法(例如最大化演化过程的累积奖励),仍是一个重要并且尚未充分研究的领域。

10.7.1 基于模型的技术

存在一种方法,其使用演化但不作为完整解决方案而是作为基于模型的方法的一个模块。在基于模型的算法中,学习器和环境的交互被用来学习一个模型,该过程要使用规划方法。当学习器从环境中收集更多的样本时模型质量不断改进,反过来又进而改进通过规划产生的策略。由于规划过程离线完成,找到良好策略所需的交互数量可以最小化,从而导致良好的在线性能。

在这种技术路线之下,规划通常通过类似值迭代的动态规划方法完成。但是,也可以使用许多其他方法。如果模型是连续或者高维的,演化或其他策略搜索方法就更受欢迎,不幸的是,绝大多数基于模型的方法通常设计为只能针对较小的、离散状态空间问题学习表格模型。然而在某些情况下,特别是大量领域知识可以获得时,更复杂的模型还是可以学习到的。

例如,线性回归已用于学习直升机的动力学模型,该模型可以用于策略搜索强化学习 [Ng et al, 2004], 学到的策略已经能够成功地控制实际的直升机模型。在最近的强化学习竞赛中,相似的方法也用于优化直升机悬停的在线学习性能,其中通过线性回归学到的模型

被用作针对策略的适应度函数, 该策略由神经演化方法离线演化 [Koppejan and Whiteson, 2009]。

另一种方法是把演化方法用于基于模型解决方案的模型学习模块。特别是, 预期学习分类器系统 (anticipatory learning classifier system) [Butz, 2002; Gerard et al, 2002, 2005; Sigaud et al, 2009] 作为一种 LCS, 可以用来演化环境的模型, 该模型可用于类似于 Dyna-Q 的框架中的规划任务 [Sutton, 1990]。

10.7.2 在线演化计算

另一种解决思路是在线演化计算 [Whiteson and Stone, 2006a,b]。主要的思想是借用时序差分方法选择动作时常用的探索方法, 并使用其选择演化过程用于评估的策略。这样一来, 就允许演化能够平衡探索和利用两部分从而优化在线性能。

当然, 演化方法已经致力于平衡探索和利用这两个部分。实际上, 这是当初启动遗传算法的主要动机之一 [Holland, 1975]。但是, 这种平衡一般在代间发生, 而不是在代内。一旦一代的个体已经确定, 这些个体通常拥有相同的评估时间。

这种方法在确定域是有意义的, 其中, 种群中每个成员可以在一段训练时段内准确评估。但是, 许多实际领域具有随机性, 其中的适应度函数需要在多个训练时段上进行平均。在这些领域内, 每个成员拥有相同的评估时间导致严重次优的动作, 这是因为在一代之内只有纯粹的探索动作。

[346]

与上面不同的是, 在线演化计算利用早先获得的信息, 系统化地给出更多评估从而获得更好的策略并且避免重新评估较差的策略。可以这样实现, 即使用时序差分的探索方法选择在每一代进行评估的策略。

例如, ϵ 贪婪选择可以用在每个训练时段开始的时候去选择评估策略。不必在人口中迭代, 演化方法以概率 ϵ 随机选择策略。该算法以概率 $1-\epsilon$ 选择当前一代中迄今为止发现的最优策略。每个策略的适应度就是当前训练时段的平均奖励。每当一个策略被选中进行评估时, 其总奖励被并入前述平均值, 并引起最优策略排名的起伏变化。

大体来说, ϵ 贪婪选择不会改变演化搜索, 只是在其中交织了利用性的训练片段, 从而增加学习过程中的平均奖励。然而, 也可以使用 softmax 选择其侧重于探索最有可能的其他解。在每一代开始时, 每个个体都被评估, 从而初始化一个训练时段的适应度。接下来, 在后续的 $e - |P|$ 个训练时段根据玻尔兹曼分布进行分配。

ϵ 贪婪和 softmax 都不能考虑估计本身的不确定性, 而这些估计是选择的基础。该缺点能够用间隔估计处理 [Kaelbling, 1993]。当用于时序差分时, 间隔估计为每一个可用动作计算置信区间为 $(100-\alpha)\%$ 的价值。学习器总是采用该区间上界最高的动作。该策略倾向于选择估计价值较高的动作, 并侧重于探索有希望但是不确定的动作。其中的 α 参数控制探索和利用的平衡点, 其较小值导致较大比重的探索。

同样的方法可以用于演化内部选择评估的策略。在每一代开始时, 每个个体为当前训练时段进行评估, 形成适应度初值。此后, 剩余的 $e - |P|$ 个训练时段将被分配给目前置信区间上界最高的策略。

以上三种在线演化计算的实现方法在与 NEAT [Whiteson and Stone, 2006b; Cardamone et al, 2009, 2010] 和 NEAT+Q [Whiteson and Stone, 2006a] 联合使用时, 能够大幅度提高在线性能。

10.8 总结

演化方法是处理具有高度挑战性的强化学习问题的利器, 特别适合于具有部分可观察性、连续动作空间和无法手工给定有效表征的问题。特别是在神经演化领域, 已经具备演化神经网络拓扑、根据网络结构分解任务和利用间接编码的复杂方法。由于混合方法, 演化计算不必放弃时序差分的能力, 不仅如此, 协同演化方法把演化计算的范围扩展到协作式和竞争式多学习器强化学习。虽然多数演化计算侧重于离线配置, 在线强化学习已经出现大有希望的研究结果, 而在线学习将是未来具有关键性且令人兴奋的挑战。

[347]

致谢。感谢 Ken Stanley、Risto Miikkulainen、Jürgen Schmidhuber、Martin Butz、Julian Bishop 以及匿名评论家对强化学习中有关演化的最新技术的宝贵意见。

参考文献

- Ackley, D., Littman, M.: Interactions between learning and evolution. *Artificial Life II*, SFI Studies in the Sciences of Complexity 10, 487–509 (1991)
- Agogino, A.K., Tumer, K.: Efficient evaluation functions for evolving coordination. *Evolutionary Computation* 16(2), 257–288 (2008)
- Arita, T., Suzuki, R.: Interactions between learning and evolution: The outstanding strategy generated by the Baldwin Effect. *Artificial Life* 7, 196–205 (2000)
- Baldwin, J.M.: A new factor in evolution. *The American Naturalist* 30, 441–451 (1896)
- Boers, E., Borst, M., Sprinkhuizen-Kuyper, I.: Evolving Artificial Neural Networks using the “Baldwin Effect”. In: *Proceedings of the International Conference Artificial Neural Nets and Genetic Algorithms in Ales, France* (1995)
- Bonarini, A.: An introduction to learning fuzzy classifier systems. *Learning Classifier Systems*, 83–104 (2000)
- Bull, L., Kovacs, T.: Foundations of learning classifier systems: An introduction. *Foundations of Learning Classifier Systems*, 1–17 (2005)
- Bull, L., O’Hara, T.: Accuracy-based neuro and neuro-fuzzy classifier systems. In: *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 905–911 (2002)
- Butz, M.: *Anticipatory learning classifier systems*. Kluwer Academic Publishers (2002)
- Butz, M.: *Rule-based evolutionary online learning systems: A principled approach to LCS analysis and design*. Springer, Heidelberg (2006)
- Butz, M., Herbot, O.: Context-dependent predictions and cognitive arm control with XCSF. In: *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pp. 1357–1364. ACM (2008)
- Butz, M., Lanzi, P.: Sequential problems that test generalization in learning classifier systems. *Evolutionary Intelligence* 2(3), 141–147 (2009)
- Butz, M., Goldberg, D., Lanzi, P.: Gradient descent methods in learning classifier systems: Improving XCS performance in multistep problems. *IEEE Transactions on Evolutionary Computation* 9(5) (2005)
- Butz, M., Lanzi, P., Wilson, S.: Function approximation with XCS: Hyperellipsoidal conditions, recursive least squares, and compaction. *IEEE Transactions on Evolutionary Computation* 12(3), 355–376 (2008)
- Butz, M., Pedersen, G., Stalph, P.: Learning sensorimotor control structures with XCSF: Redundancy exploitation and dynamic control. In: *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pp. 1171–1178 (2009)
- Cai, Z., Peng, Z.: Cooperative coevolutionary adaptive genetic algorithm in path planning of cooperative multi-mobile robot systems. *Journal of Intelligent and Robotic Systems* 33(1), 61–71 (2002)
- Cardamone, L., Loiacono, D., Lanzi, P.: On-line neuroevolution applied to the open racing car simulator. In: *Proceedings of the Congress on Evolutionary Computation (CEC)*, pp.

- 2622–2629 (2009)
- Cardamone, L., Loiacono, D., Lanzi, P.L.: Learning to drive in the open racing car simulator using online neuroevolution. *IEEE Transactions on Computational Intelligence and AI in Games* 2(3), 176–190 (2010)
- Chellapilla, K., Fogel, D.: Evolving an expert checkers playing program without using human expertise. *IEEE Transactions on Evolutionary Computation* 5(4), 422–428 (2001)
- Coello, C., Lamont, G., Van Veldhuizen, D.: *Evolutionary algorithms for solving multi-objective problems*. Springer, Heidelberg (2007)
- D'Ambrosio, D., Lehman, J., Risi, S., Stanley, K.O.: Evolving policy geometry for scalable multiagent learning. In: *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pp. 731–738 (2010)
- Darwen, P., Yao, X.: Automatic modularization by speciation. In: *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC 1996)*, pp. 88–93 (1996)
- Dasgupta, D., McGregor, D.: Designing application-specific neural networks using the structured genetic algorithm. In: *Proceedings of the International Conference on Combinations of Genetic Algorithms and Neural Networks*, pp. 87–96 (1992)
- Dawkins, R., Krebs, J.: Arms races between and within species. *Proceedings of the Royal Society of London Series B, Biological Sciences* 205(1161), 489–511 (1979)
- de Jong, E.D.: The Incremental Pareto-coevolution Archive. In: Deb, K., et al. (eds.) *GECCO 2004*. LNCS, vol. 3102, pp. 525–536. Springer, Heidelberg (2004)
- de Jong, E.: A monotonic archive for Pareto-coevolution. *Evolutionary Computation* 15(1), 61–93 (2007)
- de Jong, K., Spears, W.: An analysis of the interacting roles of population size and crossover in genetic algorithms. In: *Parallel Problem Solving from Nature*, pp. 38–47 (1991)
- de Jong, K., Spears, W., Gordon, D.: Using genetic algorithms for concept learning. *Machine learning* 13(2), 161–188 (1993)
- Deb, K.: *Multi-objective optimization using evolutionary algorithms*. Wiley (2001)
- Dorigo, M., Colombetti, M.: *Robot shaping: An experiment in behavior engineering*. The MIT Press (1998)
- Downing, K.L.: Reinforced genetic programming. *Genetic Programming and Evolvable Machines* 2(3), 259–288 (2001)
- Doya, K.: Reinforcement learning in continuous time and space. *Neural Computation* 12(1), 219–245 (2000)
- Drugowitsch, J.: *Design and analysis of learning classifier systems: A probabilistic approach*. Springer, Heidelberg (2008)
- Ficici, S., Pollack, J.: A game-theoretic approach to the simple coevolutionary algorithm. In: *Parallel Problem Solving from Nature PPSN VI*, pp. 467–476. Springer, Heidelberg (2000)
- Ficici, S., Pollack, J.: Pareto optimality in coevolutionary learning. *Advances in Artificial Life*, 316–325 (2001)
- Floreano, D., Mondada, F.: Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 26(3), 396–407 (2002)
- Floreano, D., Urzelai, J.: Evolution of plastic control networks. *Autonomous Robots* 11(3), 311–317 (2001)
- French, R., Messinger, A.: Genes, phenes and the Baldwin effect: Learning and evolution in a simulated population. *Artificial Life* 4, 277–282 (1994)
- Gaskett, C., Wettergreen, D., Zelinsky, A.: Q-learning in continuous state and action spaces. *Advanced Topics in Artificial Intelligence*, 417–428 (1999)
- Gauci, J., Stanley, K.O.: A case study on the critical role of geometric regularity in machine learning. In: *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, AAAI 2008 (2008)
- Gauci, J., Stanley, K.O.: Autonomous evolution of topographic regularities in artificial neural networks. *Neural Computation* 22(7), 1860–1898 (2010)
- Gerard, P., Stolzmann, W., Sigaud, O.: YACS: a new learning classifier system using anticipation. *Soft Computing-A Fusion of Foundations, Methodologies and Applications* 6(3),

- 216–228 (2002)
- Gerard, P., Meyer, J., Sigaud, O.: Combining latent learning with dynamic programming in the modular anticipatory classifier system. *European Journal of Operational Research* 160(3), 614–637 (2005)
- Giraud-Carrier, C.: Unifying learning with evolution through Baldwinian evolution and Lamarckism: A case study. In: *Proceedings of the Symposium on Computational Intelligence and Learning (CoIL 2000)*, pp. 36–41 (2000)
- Goldberg, D.: *Genetic Algorithms in Search*. In: *Optimization and Machine Learning*, Addison-Wesley (1989)
- Goldberg, D., Deb, K.: A comparative analysis of selection schemes used in genetic algorithms. *Foundations of genetic algorithms* 1, 69–93 (1991)
- Goldberg, D., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: *Proceedings of the Second International Conference on Genetic Algorithms and their Application*, p. 49 (1987)
- Gomez, F., Miikkulainen, R.: Solving non-Markovian control tasks with neuroevolution. In: *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1356–1361 (1999)
- Gomez, F., Miikkulainen, R.: Active guidance for a finless rocket using neuroevolution. In: *GECCO 2003: Proceedings of the Genetic and Evolutionary Computation Conference* (2003)
- Gomez, F., Schmidhuber, J.: Co-evolving recurrent neurons learn deep memory POMDPs. In: *GECCO 2005: Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 491–498 (2005a)
- Gomez, F.J., Schmidhuber, J.: Evolving Modular Fast-Weight Networks for Control. In: Duch, W., Kacprzyk, J., Oja, E., Zadrozny, S. (eds.) *ICANN 2005. LNCS*, vol. 3697, pp. 383–389. Springer, Heidelberg (2005b)
- Gomez, F.J., Schmidhuber, J., Miikkulainen, R.: Efficient Non-Linear Control Through Neuroevolution. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) *ECML 2006. LNCS (LNAI)*, vol. 4212, pp. 654–662. Springer, Heidelberg (2006)
- Gomez, F., Schmidhuber, J., Miikkulainen, R.: Accelerated neural evolution through cooperatively coevolved synapses. *Journal of Machine Learning Research* 9, 937–965 (2008)
- Gruau, F.: Automatic definition of modular neural networks. *Adaptive Behavior* 3(2), 151 (1994)
- Gruau, F., Whitley, D.: Adding learning to the cellular development of neural networks: Evolution and the Baldwin effect. *Evolutionary Computation* 1, 213–233 (1993)
- Hansen, N., Müller, S., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation* 11(1), 1–18 (2003)
- van Hasselt, H., Wiering, M.: Reinforcement learning in continuous action spaces. In: *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning, ADPRL*, pp. 272–279 (2007)
- Haykin, S.: *Neural networks: a comprehensive foundation*. Prentice-Hall (1994)
- Heidrich-Meisner, V., Igel, C.: Variable metric reinforcement learning methods applied to the noisy mountain car problem. *Recent Advances in Reinforcement Learning*, 136–150 (2008)
- Heidrich-Meisner, V., Igel, C.: Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In: *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 401–408 (2009a)
- Heidrich-Meisner, V., Igel, C.: Neuroevolution strategies for episodic reinforcement learning. *Journal of Algorithms* 64(4), 152–168 (2009b)
- Heidrich-Meisner, V., Igel, C.: Uncertainty handling CMA-ES for reinforcement learning. In: *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pp. 1211–1218 (2009c)
- Hillis, W.: Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena* 42(1-3), 228–234 (1990)
- Hinton, G.E., Nowlan, S.J.: How learning can guide evolution. *Complex Systems* 1, 495–502

- (1987)
- Holland, J., Reitman, J.: Cognitive systems based on adaptive algorithms. *ACM SIGART Bulletin* 63, 49–49 (1977)
- Holland, J.H.: *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology*. In: *Control and Artificial Intelligence*. University of Michigan Press (1975)
- Hornby, G., Pollack, J.: Creating high-level components with a generative representation for body-brain evolution. *Artificial Life* 8(3), 223–246 (2002)
- Igel, C.: Neuroevolution for reinforcement learning using evolution strategies. In: *Congress on Evolutionary Computation*, vol. 4, pp. 2588–2595 (2003)
- Jansen, T., Wiegand, R.P.: The cooperative coevolutionary (1+1) EA. *Evolutionary Computation* 12(4), 405–434 (2004)
- Kaelbling, L.P.: *Learning in Embedded Systems*. MIT Press (1993)
- Kernbach, S., Meister, E., Scholz, O., Humza, R., Liedke, J., Ricotti, L., Jemai, J., Havlik, J., Liu, W.: Evolutionary robotics: The next-generation-platform for on-line and on-board artificial evolution. In: *CEC 2009: IEEE Congress on Evolutionary Computation*, pp. 1079–1086 (2009)
- Kohl, N., Mikkilainen, R.: Evolving neural networks for fractured domains. In: *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1405–1412 (2008)
- Kohl, N., Mikkilainen, R.: Evolving neural networks for strategic decision-making problems. *Neural Networks* 22, 326–337 (2009); (special issue on Goal-Directed Neural Systems)
- Koppejan, R., Whiteson, S.: Neuroevolutionary reinforcement learning for generalized helicopter control. In: *GECCO 2009: Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 145–152 (2009)
- Kovacs, T.: *Strength or accuracy: credit assignment in learning classifier systems*. Springer, Heidelberg (2003)
- Larranaga, P., Lozano, J.: *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer, Netherlands (2002)
- Lindenmayer, A.: Mathematical models for cellular interactions in development II. Simple and branching filaments with two-sided inputs. *Journal of Theoretical Biology* 18(3), 300–315 (1968)
- Littman, M.L., Dean, T.L., Kaelbling, L.P.: On the complexity of solving Markov decision processes. In: *Proceedings of the Eleventh International Conference on Uncertainty in Artificial Intelligence*, pp. 394–402 (1995)
- Lucas, S.M., Runarsson, T.P.: Temporal difference learning versus co-evolution for acquiring othello position evaluation. In: *IEEE Symposium on Computational Intelligence and Games* (2006)
- Lucas, S.M., Togelius, J.: Point-to-point car racing: an initial study of evolution versus temporal difference learning. In: *Symposium, I.E.E.E. (ed.) on Computational Intelligence and Games*, pp. 260–267 (2007)
- Mahadevan, S., Maggioni, M.: Proto-value functions: A Laplacian framework for learning representation and control in Markov decision processes. *Journal of Machine Learning Research* 8, 2169–2231 (2007)
- Mahfoud, S.: A comparison of parallel and sequential niching methods. In: *Conference on Genetic Algorithms*, vol. 136, p. 143 (1995)
- McQuesten, P., Mikkilainen, R.: Culling and teaching in neuro-evolution. In: *Proceedings of the Seventh International Conference on Genetic Algorithms*, pp. 760–767 (1997)
- Meyer, J., Husbands, P., Harvey, I.: Evolutionary robotics: A survey of applications and problems. In: *Evolutionary Robotics*, pp. 1–21. Springer, Heidelberg (1998)
- Millán, J., Posenato, D., Dedieu, E.: Continuous-action Q-learning. *Machine Learning* 49(2), 247–265 (2002)
- Monroy, G., Stanley, K., Mikkilainen, R.: Coevolution of neural networks using a layered Pareto archive. In: *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, p. 336 (2006)
- Moriarty, D., Mikkilainen, R.: Forming neural networks through efficient and adaptive

- coevolution. *Evolutionary Computation* 5(4), 373–399 (1997)
- Moriarty, D.E., Miikkulainen, R.: Efficient reinforcement learning through symbiotic evolution. *Machine Learning* 22(11), 11–33 (1996)
- Moriarty, D.E., Schultz, A.C., Grefenstette, J.J.: Evolutionary algorithms for reinforcement learning. *Journal of Artificial Intelligence Research* 11, 199–229 (1999)
- Ng, A.Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., Liang, E.: Inverted autonomous helicopter flight via reinforcement learning. In: *Proceedings of the International Symposium on Experimental Robotics* (2004)
- Nolfi, S., Parisi, D.: Learning to adapt to changing environments in evolving neural networks. *Adaptive Behavior* 5(1), 75–98 (1997)
- Nolfi, S., Elman, J.L., Parisi, D.: Learning and evolution in neural networks. *Adaptive Behavior* 2, 5–28 (1994)
- Nordin, P., Banzhaf, W.: An on-line method to evolve behavior and to control a miniature robot in real time with genetic programming. *Adaptive Behavior* 5(2), 107 (1997)
- Panait, L., Luke, S.: Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multi-Agent Systems* 11(3), 387–434 (2005)
- Panait, L., Luke, S., Harrison, J.F.: Archive-based cooperative coevolutionary algorithms. In: *GECCO 2006: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pp. 345–352 (2006)
- Parr, R., Painter-Wakefield, C., Li, L., Littman, M.: Analyzing feature generation for value-function approximation. In: *Proceedings of the 24th International Conference on Machine Learning*, p. 744 (2007)
- Pereira, F.B., Costa, E.: Understanding the role of learning in the evolution of busy beaver: A comparison between the Baldwin Effect and a Lamarckian strategy. In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2001* (2001)
- Peters, J., Schaal, S.: Natural actor-critic. *Neurocomputing* 71(7-9), 1180–1190 (2008)
- Pollack, J., Blair, A.: Co-evolution in the successful learning of backgammon strategy. *Machine Learning* 32(3), 225–240 (1998)
- Popovici, E., Bucci, A., Wiegand, P., De Jong, E.: Coevolutionary principles. In: Rozenberg, G., Baeck, T., Kok, J. (eds.) *Handbook of Natural Computing*. Springer, Berlin (2010)
- Potter, M.A., De Jong, K.A.: Evolving neural networks with collaborative species. In: *Summer Computer Simulation Conference*, pp. 340–345 (1995)
- Potter, M.A., De Jong, K.A.: Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation* 8, 1–29 (2000)
- Pratihari, D.: Evolutionary robotics: A review. *Sadhana* 28(6), 999–1009 (2003)
- Priesterjahn, S., Weimer, A., Eberling, M.: Real-time imitation-based adaptation of gaming behaviour in modern computer games. In: *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1431–1432 (2008)
- Radcliffe, N.: Genetic set recombination and its application to neural network topology optimisation. *Neural Computing & Applications* 1(1), 67–90 (1993)
- Rosin, C.D., Belew, R.K.: New methods for competitive coevolution. *Evolutionary Computation* 5(1), 1–29 (1997)
- Rubinstein, R., Kroese, D.: The cross-entropy method: a unified approach to combinatorial optimization. In: *Monte-Carlo Simulation, and Machine Learning*. Springer, Heidelberg (2004)
- Runarsson, T.P., Lucas, S.M.: Co-evolution versus self-play temporal difference learning for acquiring position evaluation in small-board go. *IEEE Transactions on Evolutionary Computation* 9, 628–640 (2005)
- Schmidhuber, J., Wierstra, D., Gomez, F.J.: Evolino: Hybrid neuroevolution / optimal linear search for sequence learning. In: *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pp. 853–858 (2005)
- Schmidhuber, J., Wierstra, D., Gagliolo, M., Gomez, F.: Training recurrent networks by evolino. *Neural Computation* 19(3), 757–779 (2007)
- Schroder, P., Green, B., Grum, N., Fleming, P.: On-line evolution of robust control systems: an industrial active magnetic bearing application. *Control Engineering Practice* 9(1), 37–49 (2001)

- Sigaud, O., Butz, M., Kozlova, O., Meyer, C.: Anticipatory Learning Classifier Systems and Factored Reinforcement Learning. *Anticipatory Behavior in Adaptive Learning Systems*, 321–333 (2009)
- Stanley, K., Miikkulainen, R.: A taxonomy for artificial embryogeny. *Artificial Life* 9(2), 93–130 (2003)
- Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. *Evolutionary Computation* 10(2), 99–127 (2002)
- Stanley, K.O., Miikkulainen, R.: Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research* 21, 63–100 (2004a)
- Stanley, K.O., Miikkulainen, R.: Evolving a Roving Eye for Go. In: Deb, K., et al. (eds.) *GECCO 2004*. LNCS, vol. 3103, pp. 1226–1238. Springer, Heidelberg (2004b)
- Stanley, K.O., Bryant, B.D., Miikkulainen, R.: Evolving adaptive neural networks with and without adaptive synapses. In: *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*, vol. 4, pp. 2557–2564 (2003)
- Stanley, K.O., D'Ambrosio, D.B., Gauci, J.: A hypercube-based indirect encoding for evolving large-scale neural networks. *Artificial Life* 15(2), 185–212 (2009)
- Steels, L.: Emergent functionality in robotic agents through on-line evolution. In: *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pp. 8–16 (1994)
- Sutton, R.S.: Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In: *Proceedings of the Seventh International Conference on Machine Learning*, pp. 216–224 (1990)
- Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press (1998)
- Sywerda, G.: Uniform crossover in genetic algorithms. In: *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 2–9 (1989)
- Tan, C., Ang, J., Tan, K., Tay, A.: Online adaptive controller for simulated car racing. In: *Congress on Evolutionary Computation (CEC)*, pp. 2239–2245 (2008)
- Taylor, M.E., Whiteson, S., Stone, P.: Comparing evolutionary and temporal difference methods in a reinforcement learning domain. In: *GECCO 2006: Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1321–1328 (2006)
- Tesauro, G.: TD-gammon, a self-teaching backgammon program achieves master-level play. *Neural Computation* 6, 215–219 (1994)
- Tesauro, G.: Comments on co-evolution in the successful learning of backgammon strategy. *Machine Learning* 32(3), 241–243 (1998)
- Verbancsics, P., Stanley, K.: Evolving Static Representations for Task Transfer. *Journal of Machine Learning Research* 11, 1737–1769 (2010)
- Von Neumann, J.: Zur Theorie der Gesellschaftsspiele *Math. Annalen* 100, 295–320 (1928)
- Whiteson, S., Stone, P.: Evolutionary function approximation for reinforcement learning. *Journal of Machine Learning Research* 7, 877–917 (2006a)
- Whiteson, S., Stone, P.: On-line evolutionary computation for reinforcement learning in stochastic domains. In: *GECCO 2006: Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1577–1584 (2006b)
- Whiteson, S., Kohl, N., Miikkulainen, R., Stone, P.: Evolving keepaway soccer players through task decomposition. *Machine Learning* 59(1), 5–30 (2005)
- Whiteson, S., Tanner, B., White, A.: The reinforcement learning competitions. *AI Magazine* 31(2), 81–94 (2010a)
- Whiteson, S., Taylor, M.E., Stone, P.: Critical factors in the empirical performance of temporal difference and evolutionary methods for reinforcement learning. *Autonomous Agents and Multi-Agent Systems* 21(1), 1–27 (2010b)
- Whitley, D., Dominic, S., Das, R., Anderson, C.W.: Genetic reinforcement learning for neurocontrol problems. *Machine Learning* 13, 259–284 (1993)
- Whitley, D., Gordon, S., Mathias, K.: Lamarckian evolution, the Baldwin effect and function optimization. In: *Parallel Problem Solving from Nature - PPSN III*, pp. 6–15 (1994)
- Wiegand, R., Liles, W., De Jong, K.: An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 1235–1242 (2001)

- Wieland, A.: Evolving neural network controllers for unstable systems. In: International Joint Conference on Neural Networks, vol 2, pp. 667–673 (1991)
- Wilson, S.: Classifier fitness based on accuracy. *Evolutionary Computation* 3(2), 149–175 (1995)
- Wilson, S.: Function approximation with a classifier system. In: GECCO 2001: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 974–982 (2001)
- Wolpert, D., Tumer, K.: Optimal payoff functions for members of collectives. *Modeling Complexity in Economic and Social Systems*, 355 (2002)
- Yamasaki, K., Sekiguchi, M.: Clear explanation of different adaptive behaviors between Darwinian population and Lamarckian population in changing environment. In: Proceedings of the Fifth International Symposium on Artificial Life and Robotics, vol. 1, pp. 120–123 (2000)
- Yao, X.: Evolving artificial neural networks. *Proceedings of the IEEE* 87(9), 1423–1447 (1999)
- Yong, C.H., Miikkulainen, R.: Coevolution of role-based cooperation in multi-agent systems. Tech. Rep. AI07-338, Department of Computer Sciences, The University of Texas at Austin (2007)
- Zhang, B., Muhlenbein, H.: Evolving optimal neural networks using genetic algorithms with Occam's razor. *Complex Systems* 7(3), 199–220 (1993)
- Zufferey, J.-C., Floreano, D., van Leeuwen, M., Merenda, T.: Evolving vision-based flying robots. In: Bülthoff, H.H., Lee, S.-W., Poggio, T.A., Wallraven, C. (eds.) *BMCV 2002. LNCS*, vol. 2525, pp. 592–600. Springer, Heidelberg (2002)

概率模型

贝叶斯强化学习

Nikos Vlassis, Mohammad Ghavamzadeh, Shie Mannor, Pascal Poupart

摘要

本章综述利用贝叶斯技术进行强化学习的近期工作。在贝叶斯学习中，不确定性由基于未知参数的先验分布表示，学习通过根据观测数据计算后验分布而实现。因此，与其他强化学习形式的显著区别在于，贝叶斯强化学习为诸如模型参数、价值函数、策略和梯度的各种参数维护各组的概率分布。这一特点带来若干优势：1) 领域知识可以自然地由先验概率分布编码，从而加速学习，2) 能够自然地优化探索 / 利用的折衷；3) 在获得鲁棒策略的过程中可以很自然地考虑风险。

11.1 简介

359

贝叶斯强化学习很可能是最古老的强化学习技术。早在 1950 和 1960，几位运筹学研究人員就已经研究了通过不确定概率控制马尔可夫链的问题。贝尔曼开发了用于贝叶斯赌博机问题 (Bayesian bandit problem) 的动态规划技术 [Bellman, 1956 ; Bellman and Kalaba, 1959 ; Bellman, 1961]。该研究之后被推广到拥有未知迁移概率和奖励的多状态决策问题 [Silver, 1963 ; Cozzolino, 1964 ; Cozzolino et al, 1965]。Martin 撰写的《贝叶斯决策问题和马尔可夫链》(Bayesian Decision Problems and Markov Chains) 对当时的工作进行了综述 [Martin 1967]。那时，强化学习称为“自适应控制过程”(adaptive control processes) 以及“贝叶斯自适应控制”(Bayesian adaptive control)。

由于贝叶斯学习与决策理论紧密结合，贝叶斯技术是同时学习环境并做出决策的自然候选技术。其思想是把未知参数处理为随机变量，并为这些变量维护一个显式的分布从而量化不确定性。在证据不断积累的过程中，以上分布被不断更新，而决策就依赖于消除掉(通过积分)未知参数。

对比于通常学习参数的点分布的传统强化学习技术，显式分布允许对不确定性进行量化，从而能够加速学习并减少风险。特别是，先验分布允许分析人员对领域知识进行编码并减少不确定性。对大多数真实世界问题来说，由于在迁移、观测和奖励函数全部未知时太多的参数需要学习，完全从零开始的强化学习的计算量过大。因此，通过在先验分布中编码领域知识，与环境交互从而找到优秀策略的数量可以大幅度减少。不仅如此，领域知识可以帮助避免灾难性事件，否则执行事件需要被反复学习。参数的显式分布也提供了不确定性的量化表征，这有助于优化探索和利用的折衷。动作的选择通常通过最大化模型现有估计(利用)的未来奖励而得到，但是有必要探索模型的不确定部分从而对之进行精炼并在未来获得更大奖励。因此，这种由显式分布表征的量化未来不确定性非常有用。同样的，对未来回报的不确定性的显式量化也有助于减少奖励的起伏并减少低回报风险。

本章组织如下。11.2 节表述针对无模型强化学习的贝叶斯技术，其中价值函数、策略和梯度的参数拥有显式分布。11.3 节介绍针对基于模型的强化学习的贝叶斯技术，其中迁移、观测和奖励函数的参数均有显式分布。最后，11.4 节讨论考虑有限样本的贝叶斯技术，从而获得样本复杂性上下界的估计并在不确定性下进行优化。

360

11.2 无模型贝叶斯强化学习

无模型强化学习不显式学习系统模型，而是使用与系统直接交互的样本轨迹。无模型技术通常更易于实现，因为其不需要任何表征模型的数据结构以及相应的模型更新算法。但是，对无模型技术进行推理通常会更加复杂，这是因为不是总能够清楚地知道需要多少个样本轨迹才能更新最优策略和价值函数的估计。本节中，我们描述几种贝叶斯技术，其中将价值函数和策略梯度处理为遵循某种分布的随机变量。具体说来，11.2.1 节介绍学习 Q 函数分布的方法，11.2.2 节考虑策略梯度的分布，11.2.3 节讨论怎样使用价值函数的分布去推断演员-评论家算法策略梯度的分布。

11.2.1 基于价值函数的算法

基于价值函数的强化学习方法搜索价值函数空间从而找到最优价值（动作-值）函数，并用其找到最优策略。在本小节里，我们讨论两种基于价值函数的贝叶斯强化学习算法：贝叶斯 Q 学习 [Dearden et al, 1998] 和高斯过程时序差分学习 [Engel et al, 2003, 2005a; Engel, 2005]。第一种算法适合离散状态域和动作空间，第二种算法处理连续状态和动作空间。

11.2.1.1 贝叶斯 Q 学习

贝叶斯 Q 学习 (BQL) [Dearden et al, 1998] 是针对广泛使用的 Q 学习算法 [Watkins, 1989] 的贝叶斯技术，其中探索和利用通过显式维护一个选择动作的 Q 值分布来加以平衡。用 $D(s,a)$ 表示一个随机变量，其表征在状态 s 采用动作 a 并跟随其后的最优策略的折扣奖励之和。该变量的期望 $\mathbb{E}[D(s,a)] = Q(s,a)$ 是经典的 Q 函数。在 BQL 中，为在任意状态 $s \in \mathcal{S}$ 采用任意动作 $a \in \mathcal{A}$ 的 $D(s,a)$ 设置一个先验概率，当观测到 $D(s,a)$ 的独立样本时更新其后验概率。BQL 的目的是通过减少 $\mathbb{E}[D(s,a)]$ 的不确定性来学习 $Q(s,a)$ 。BQL 做了以下简化假设：1) $D(s,a)$ 遵循正态分布，并拥有均值 $\mu(s,a)$ 和精度 $\tau(s,a)$ [⊖]。该假设意味着，建立均值 $\mu(s,a)$ 和精度 $\tau(s,a)$ 的模型足以描述分布 $D(s,a)$ 的不确定性。2) 假定针对状态-动作对 (s,a) 先验概率分布 $P(D(s,a))$ 互相独立，并且是正态-伽马分布。该假定对系统先验知识的形式进行约束，但是保证在给定采样获得的折扣奖励和 $d = \sum_t \gamma^t r(s_t, a_t)$ 的条件下，其后验概率 $P(D(s,a)|d)$ 也是正态-伽马分布。但是，由于不同状态-动作对 (s,a) 的折扣奖励和满足贝尔曼方程，因此其后验概率变为互相关联（即不独立）。3) 为了确保形式上的简单，后验概率被强制为独立（打破相关性）。

361

在 BQL 中，我们存储每个 $D(s,a)$ 上分布的超参数 (hyper-parameter)，而不是像标准 Q 学习那样存储 Q 值。因此，BQL 在其最原始的形式上只能用于有限状态和动作空间的 MDP 问题。在每个时间步长上，在状态 s 执行 a 后如果观测到 r 和 s' ，则 D 上的分布和更新规则如下：

⊖ 高斯随机变量的精度是其方差的倒数。

$$P(D(s,a)|r,s') = \int_a P(D(s,a)|r+\gamma d)P(D(s',a')=d) \\ \propto \int_a P(D(s,a))P(r+\gamma d|D(s,a))P(D(s',a')=d)$$

由于后验概率在积分时没有闭合解, 因此通过最小化 KL 散度 (KL-divergence) 来寻找正态-伽马分布的近似解。

在执行过程中, 选择具有 Q 值最高期望 (即 $a^* = \operatorname{argmax}_a \mathbb{E}[Q(s,a)]$) 的动作当然是很诱人的, 但是这种策略并不保证利用。为了解决这个问题, [Dearden et al, 1998] 提出把利用红利添加到期望 Q 值, 从而估计近期的完美信息价值 (Value of Perfect Information, VPI)。

$$a^* = \operatorname{argmax}_a \mathbb{E}[Q(s,a)] + VPI(s,a)$$

如果利用导致策略的改变, 则应该考虑其价值函数的增益。由于学习器并不事先知道采取动作后的效果, VPI 计算为增益的期望:

$$VPI(s,a) = \int_{-\infty}^{\infty} dx \operatorname{Gain}_{s,a}(x)P(Q(s,a)=x) \quad (11.1)$$

其中增益对应于学习该动作的精确 Q 值 (记作 $q_{s,a}$) 而导致的改进。

$$\operatorname{Gain}_{s,a}(q_{s,a}) = \begin{cases} q_{s,a} - \mathbb{E}[Q(s,a_1)] & , a \neq a_1 \text{ 且 } q_{s,a} > \mathbb{E}[Q(s,a_1)] \\ \mathbb{E}[Q(s,a_2)] - q_{s,a} & , a = a_1 \text{ 且 } q_{s,a} < \mathbb{E}[Q(s,a_2)] \\ 0 & , \text{其他} \end{cases} \quad (11.2)$$

这里有两种情况: 相对于拥有最高期望 Q 值的动作 a_1 , 动作 a 拥有更高实际 Q 值; 或者, 拥有最高期望 Q 值的动作 a_1 反而比拥有第二高期望 Q 值的动作 a_2 的实际 Q 值低。

11.2.1.2 高斯过程时序差分学习

[362]

BQL 为每个 (s,a) 对维护一个定义在 $D(s,a)$ 上的分布。因此, BQL 不能用于连续状态和动作空间问题。[Engel et al, 2003a, 2005a] 提出了使用高斯过程的自然延伸。类似于 BQL, 假定 $D(s,a)$ 是均值为 $\mu(s,a)$ 和精度为 $\tau(s,a)$ 的正态分布。但是, 不需要维护同时定义在 μ 和 τ 上的正态-伽马分布, 而只需要建立在 μ 上的高斯分布。由于 $\mu(s,a) = Q(s,a)$, 并且需要学习的主要数量是 Q 函数, 因此可以只维护对于均值的信心。为了能够处理无限状态和动作空间, 可以使用高斯过程为针对每一个 (s,a) 对的无限多个定义在 $Q(s,a)$ 上的高斯分布建模。

高斯过程 (例如 [Rasmussen and Williams, 2006]) 是多值高斯分布向无穷多维空间的延伸, 以及等效的对由无穷多个单值高斯分布组成的联合分布的延伸。高斯过程 $GP(\mu,k)$ 由均值函数 $\mu(x)$ 和核函数 $k(x,x')$ 参数化, 分别是均值向量的极限和维度为无穷的多变量高斯分布的协方差矩阵。高斯过程常用于基于对未知底层函数的采样进行函数回归。

沿着上述方向, [Engel et al, 2003, 2005a] 提出高斯过程时序差分 (Gaussian Process Temporal Difference, GPTD) 技术去在折扣收益和的基础上学习策略的 Q 函数。回想一下针对固定策略 π 的折扣收益和的分布可以递归定义为:

$$D(z) = r(z) + \gamma D(z'), \text{ 其中 } z' \sim P^\pi(z'|z) \quad (11.3)$$

当 z 表示状态集合时, $\mathbb{E}[D] = V$, 当 z 表示状态-动作对时, $\mathbb{E}[D] = Q$ 。除非专门指出, 我们假定 $z = (s,a)$ 。可以把 D 分解为其均值之和 Q 和一个零均值噪声项 ΔQ , 后一项允许我们直接在 Q 上叠加一个概率分布。在公式 (11.3) 中用 $Q(z) + \Delta Q(z)$ 代替 $D(z)$, 并把 ΔQ 组合到一个零均值噪声项 $N(z,z') = \Delta Q(z) - \gamma \Delta Q(z')$, 可以得到:

$$r(z) = Q(z) - \gamma Q(z') + N(z,z'), \text{ 其中 } z' \sim P^\pi(z'|z) \quad (11.4)$$

GPTD 学习模型 [Engel et al, 2003, 2005a] 建立在公式 (11.4) 的统计生成模型之上, 该式

把观测奖励信号 r 和不可观测的动作 - 价值函数 Q 联系起来。现在假设观测到序列 z_0, z_1, \dots, z_t , 则公式 (11.4) 引出一个 t 个方程式的系统, 以矩阵形式表示为:

$$r_{t-1} = H_t Q_t + N_t \quad (11.5)$$

其中

$$\begin{aligned} r_t &= (r(z_0), \dots, r(z_t))^T, & Q_t &= (Q(z_0), \dots, Q(z_t))^T \\ N_t &= (N(z_0, z_1), \dots, N(z_{t-1}, z_t))^T \end{aligned} \quad (11.6)$$

363

$$H_t = \begin{bmatrix} 1 & -\gamma & 0 & \dots & 0 \\ 0 & 1 & -\gamma & \dots & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \dots & 1 & -\gamma \end{bmatrix} \quad (11.7)$$

如果假定残差 $\Delta Q(z_0), \dots, \Delta Q(z_t)$ 服从零均值高斯分布, 其方差为 σ^2 , 并且残差互相独立生成, 即 $\mathbb{E}[\Delta Q(z_i) \Delta Q(z_j)] = 0$ ($i \neq j$), 易得噪声向量 N_t 具有均值 0 和协方差矩阵:

$$\Sigma_t = \sigma^2 H_t H_t^T = \sigma^2 \begin{bmatrix} 1+\gamma^2 & -\gamma & 0 & \dots & 0 \\ -\gamma & 1+\gamma^2 & -\gamma & \dots & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & \dots & -\gamma & 1+\gamma^2 \end{bmatrix} \quad (11.8)$$

在时段式训练任务中, 如果 z_{t-1} 是该训练时段的最后一个状态 - 动作对 (即 s_t 是一个零奖励的吸收式终止状态), H_t 变成 $t \times t$ 的可逆矩阵, 其形式如公式 (11.7) (删除了最后一列)。其对于噪声协方差矩阵 Σ_t 的效果是右下角元素从 $1+\gamma^2$ 变为 1。

在 Q 上设置高斯过程先验概率 $GP(0, k)$, 我们可以使用贝叶斯规则获得 Q 上的后验高斯过程的矩 (动差) \hat{Q} 和 \hat{k} :

$$\hat{Q}_t(z) = \mathbb{E}[Q(z) | \mathcal{D}_t] = k_t(z)^T \alpha_t \quad (11.9)$$

$$\hat{k}_t(z, z') = \text{Cov}[Q(z), Q(z') | \mathcal{D}_t] = k(z, z') - k_t(z)^T C_t k_t(z')$$

其中 \mathcal{D}_t 表示至时间步长 t (包含 t) 的观测数据。这里有以下定义:

$$\begin{aligned} k_t(z) &= (k(z_0, z), \dots, k(z_{t-1}, z))^T, & K_t &= [k_t(z_0), k_t(z_1), \dots, k_t(z_t)] \\ \alpha_t &= H_t^T (H_t K_t H_t^T + \sum_i I)^{-1} r_{t-1}, & C_t &= H_t^T (H_t K_t H_t^T + \sum_i I)^{-1} H_t \end{aligned} \quad (11.10)$$

当观测到更多数据时, 后验协方差降低, 反映出对 Q 函数的估计 \hat{Q}_t 具有更高信心。

上述 GPTD 模型是基于内核的非参数模型, 也可以在相似前提下使用参数化表征。在参数化设置下, 假定高斯过程 Q 由一有限基函数 $Q(\cdot) = \varphi(\cdot)^T W$ 的线性组合构成, 其中 φ 是特征向量, W 是权重向量。在参数化 GPTD 中, Q 的随机性归因于 W 是随机向量。该模型中, 我们为 W 设置高斯先验, 并使用贝叶斯规则计算 W 在观测数据上的条件后验分布。 Q 的后验均值和协方差可以容易地通过 W 的后验矩与特征向量 φ 的乘积计算。请参阅文献 [Engel, 2005] 获得更多参数化 GPTD 的细节。

364

在参数化情况下, 后验概率的计算可以在每样本 $O(n^2)$ 的时间复杂度和每样本 $O(n^2)$ 的空间复杂度下完成, 其中 n 是用于近似 Q 的基函数的个数。在非参数化情况下, 可以为观测到的每个样本配备一个新的基函数, 使得叠加第 t 个样本的时间和空间复杂度均为 $O(n^2)$ 。这似乎说明非参数形式的 GPTD 只对简单小问题计算有效。然而, 非参数化 GPTD 的计算成本可以通过在线稀疏化方法 (例如 [Engel et al 2002]) 降低至能够有效在线计算的水平。

先验分布的选择能够严重影响 GPTD 的性能。但是, 在标准 GPTD 问题上, 先验在一开始时设置并在算法执行过程中保持不变。[Reisinger et al, 2008] 研究了一种使用顺序马

尔可夫链的在线模型选择方法,称为代替核函数强化学习(replacing-kernel RL),并在实际问题中能够比许多核函数族在标准 GPTD 上的性能更为出色。

最后, GPTD 可以用于推导 SARSA 类型的算法,称为 GPSARSA[Engel et al, 2005a; Engel, 2005],其中状态-动作对使用 GPTD 估计,而策略由 ε -贪婪方法改进直至 ε 趋于 0。GPTD 框架,特别是 GPSARSA 算法,已成功应用到大规模强化学习问题,例如章鱼手臂控制[Engel et al, 2005b]和无线网络关联控制[Aharony et al, 2005]。

11.2.2 策略梯度算法

策略梯度(Policy Gradient, PG)方法是基于参数化动作-选择策略的强化学习算法,该方法通过向性能评估的梯度方向移动参数实现参数更新(例如[Williams, 1992; Marbach, 1998, Baxter and Bartlett, 2001])。这些算法已经从理论和实际两个角度进行了分析(例如[Marbach, 1998; Baxter and Bartlett, 2001]),并扩展到 POMDP[Baxter and Bartlett, 2001]。但是,理论结果和实际评估都指出该类算法的主要缺点,即梯度估计的高度起伏。

已经有若干解决方法针对以上问题,例如 1) 在算法中使用人工阻尼系数($0 < \gamma < 1$) [Marbach, 1998; Baxter and Bartlett, 2001]。但是,这种方法由于在梯度估计中引入了偏置而导致了另一个问题。2) 在 PG 算法的更新过程中,从平均奖励的估计中减去增强基线(reinforcement baseline) [Williams, 1992; Marbach, 1998; Sutton et al, 2000; Greensmith et al, 2004]。该方法不涉及梯度估计的偏置,但是,怎样选择良好的状态相关基线几乎仍是一个没有解决的问题。3) 用所谓的自然策略梯度(natural policy gradient)取代策略梯度的估计[Kakade, 2002; Bagnell and Schneider, 2003; Peters et al, 2003]。在策略更新规则上,根据自然梯度规则的移动相当于使用策略的逆 Fisher 信息矩阵对梯度进行转换。在实践评估中,可证明自然 PG 能够大幅度超越传统 PG 的性能[Kakade, 2002; Bagnell and Schneider, 2003; Peters et al, 2003; Peters and Schaal, 2008]]。

然而,传统和自然策略梯度方法都依赖于蒙特卡罗技术对性能指标的梯度进行估计。尽管蒙特卡罗估计是无偏的,但是经常导致极大的起伏,或者需要过大的样本数量(参见[O'Hagan, 1987])。在策略梯度估计的情况下,以上问题将由于持续的策略改进要求多个梯度估计步骤这一事实而进一步恶化。[O'Hagan, 1991]提出一种蒙特卡罗积分估计的贝叶斯变体,称为贝叶斯象限(Bayesian Quadrature, BQ)。其思想是把形如 $\int dx f(x)g(x)$ 的积分处理为随机量,通过把积分第一项 f 作为一个随机函数,该函数定义了用高斯过程形式体现的先验分布。在一系列采样点 $\{x_1, x_2, \dots, x_M\}$ 观测(可能有噪声) f 的样本,就可以使用贝叶斯规则计算基于这些样本的 f 的后验分布。这一处理依次为积分值产生一个后验分布。[Rasmussen and Ghahramani, 2003]以实验方式证明这种方法应用在评估期望时显著优于蒙特卡罗估计,在均方误差指标上可以小几个数量级。有趣的是, BQ 经常在 f 已知时有效。函数 f 的后验分布可以看成 f 的近似(极限情况趋于 f),但这种近似可以用于执行闭合式的积分,对比而言,蒙特卡罗积分使用 f 的精确形式,但只能在采样点进行。因此 BQ 通过后验在样本之间内插并且执行闭合式积分更好地使用了样本提供的信息。

本节中,我们研究一个针对策略梯度估计的贝叶斯框架,该框架建立在把策略梯度建模为高斯过程的基础上[Ghavamzadeh and Engel, 2006]。这样就能够减少达到精确梯度估计所需要的样本个数,而且同时提供了自然梯度的估计以及梯度估计不确定性的度量(即梯度

协方差)。

我们从定义和符号开始。静态策略 $\pi(\cdot|s)$ 是动作空间上的概率分布, 以当前状态为条件。给定固定策略 π , MDP 引发一个在状态-动作对上的马尔可夫链, 其从 (s_t, a_t) 到 (s_{t+1}, a_{t+1}) 的迁移概率为 $\pi(a_{t+1}|s_{t+1})P(s_{t+1}|s_t, a_t)$ 。一般定义由该马尔可夫链产生的一条路径为 $\xi = (s_0, a_0, s_1, a_1, \dots, s_{T-1}, a_{T-1}, s_T)$, $T \in \{0, 1, \dots, \infty\}$ 。则该路径的概率(密度)为:

$$P(\xi|\pi) = P_0(s_0) \prod_{t=0}^{T-1} \pi(a_t|s_t) P(s_{t+1}|s_t, a_t) \quad (11.11)$$

路径 ξ 的折扣积累回报记作 $R(\xi) = \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t)$, 其中 $\gamma \in [0, 1]$ 是阻尼系数。由于 ξ 本身是一个随机变量, 而且针对指定路径采样得到的奖励仍然具有随机性, 因此 $R(\xi)$ 是随机变量。针对某一路径 ξ , 其 $R(\xi)$ 的数学期望为 $\bar{R}(\xi)$ 。最后, 定义策略 π 的期待回报为: [366]

$$\eta(\pi) = \mathbb{E}[R(\xi)] = \int d\xi \bar{R}(\xi) P(\xi|\pi) \quad (11.12)$$

在 PG 方法中, 定义一类平滑参数化随机策略 $\{\pi(\cdot|s; \theta), s \in \mathcal{S}, \theta \in \Theta\}$ 。我们可以从观测到的系统轨迹中以策略参数 θ 为变量估计期望回报的梯度, 然后通过沿着梯度方向调整参数值优化策略。我们使用下式估计期望回报的梯度:

$$\nabla \eta(\theta) = \int d\xi \bar{R}(\xi) \frac{\nabla P(\xi; \theta)}{P(\xi; \theta)} P(\xi; \theta) \quad (11.13)$$

其中 $\frac{\nabla P(\xi; \theta)}{P(\xi; \theta)} = \nabla \log P(\xi; \theta)$ 称为积分函数或似然率。由于其中初始状态分布 P_0 和状态-迁移分布 P 与策略参数 θ 独立, 可以把一条路径的积分函数用公式 (11.11) 写为^①:

$$u(\xi; \theta) = \frac{\nabla P(\xi; \theta)}{P(\xi; \theta)} = \sum_{t=0}^{T-1} \frac{\nabla \pi(a_t|s_t; \theta)}{\pi(a_t|s_t; \theta)} = \sum_{t=0}^{T-1} \nabla \log \pi(a_t|s_t; \theta) \quad (11.14)$$

解决 PG 的频率主义方法使用经典蒙特卡罗方法估计公式 (11.13) 的梯度, 该方法根据 $P(\xi; \theta)$ 产生独立同分布样本路径 ξ_1, \dots, ξ_M 并且利用蒙特卡罗方法估计梯度 $\nabla \eta(\theta)$:

$$\widehat{\nabla \eta}(\theta) = \frac{1}{M} \sum_{i=1}^M R(\xi_i) \nabla \log P(\xi_i; \theta) = \frac{1}{M} \sum_{i=1}^M R(\xi_i) \sum_{t=0}^{T_i-1} \nabla \log \pi(a_{t,i}|s_{t,i}; \theta) \quad (11.15)$$

这是无偏估计, 因此基于大数定理, 在 M 区域无限大时, $\widehat{\nabla \eta}(\theta)$ 以概率 1 无限趋近于 $\nabla \eta(\theta)$ 。

在频率主义 PG 方法中, 性能评估指标使用 $\eta(\theta)$ 。为了作为有用的性能指标, $\eta(\theta)$ 必须是策略参数 θ 的确定性函数。这可以通过对全部可能路径 ξ 和每条路径可能的积累回报取平均积累回报 $R(\xi)$ 实现, 在贝叶斯方法中, 我们还有附加的随机性来源, 即关于产生积累回报的过程之中的主观贝叶斯不确定性。定义 $\eta_B(\theta) = \int d\xi R(\xi) P(\xi; \theta)$, 其中 $\eta_B(\theta)$ 是关于贝叶斯不确定性的随机变量。我们感兴趣的是评估 $\eta_B(\theta)$ 梯度关于策略参数 θ 的后验分布, 梯度的后验平均值为:

$$\mathbb{E}[\nabla \eta_B(\theta) | \mathcal{D}_M] = \mathbb{E} \left[\int d\xi R(\xi) \frac{\nabla P(\xi; \theta)}{P(\xi; \theta)} P(\xi; \theta) | \mathcal{D}_M \right] \quad (11.16)$$

[Ghavamzadeh and Engel, 2006] 提出的贝叶斯策略梯度 (BPG) 方法中, 将估计期望回报 (公式 (11.16)) 的梯度表述为积分问题, 继而就可以使用上述 BQ 方法 [O'Hagan, 1991]。BQ 方法中, 我们需要把被积分项分为两个部分: $f(\xi; \theta)$ 和 $g(\xi; \theta)$ 。 $f(\xi; \theta)$ 被建模为高斯过程, 同

① 为了简化符号, 我们省略 ∇ 和 u 对参数 θ 的依赖性, 并在后续部分使用 ∇ 和 $u(\xi)$ 代替 ∇_θ 和 $u(\xi; \theta)$ 。

时假定 $g(\xi; \theta)$ 为已知函数。接下来, 计算以观测数据 $\mathcal{D}_M = \{\xi_1, \dots, \xi_M\}$ 为条件的梯度 $\nabla_{\eta_B}(\theta)$ 的后验矩。由于即使只针对特定的 ξ (奖励函数具有随机性), 一般也不可能精确知道 $R(\xi)$, 因此 $R(\xi)$ 应该总是属于 $f(\xi; \theta)$ 模型中高斯过程那一部分。[Ghavamzadeh and Engel, 2006] 提出了两种拆分公式 (11.16) 被积分项的不同方法, 从而形成两种不同的贝叶斯模型。参考文献 [Ghavamzadeh and Engel, 2006] 表 1 总结了这两个模型。模型 1 和模型 2 均为 f 的先验协方差使用 Fisher 类型内核, 选择该内核是受到下述观点的启发, 即好的表征方法应该依赖于数据产生过程 (详见 [Jaakkola and Haussler, 1999; Shawe-Taylor and Cristianini, 2004])。具体选择线性还是二次 Fisher 内核则由梯度后验矩必须具有适当计算复杂度的要求决定。

模型 1 和模型 2 可以用来定义算法, 从而评估期望回报关于策略参数的梯度。这些算法使用一组策略参数 θ 和一组数量为 M 的样本作为输入, 并返回对期望回报梯度的后验矩的估计。这些贝叶斯策略梯度评估算法进而可以导向贝叶斯策略梯度 (BPG) 算法。该算法始于一组策略参数矢量 θ_0 , 并在通过贝叶斯 PG 评估方法计算的期望回报梯度的后验均值方向上更新参数。该过程重复 N 次, 或者直至梯度估计足够接近于 0。

前面已经提到过, 模型 1 和模型 2 使用的内核函数均基于 Fisher 信息矩阵 $G(\theta)$ 。于是, 每次更新策略参数时需要重新计算 G 。在大多数实际场合下, G 未知并需要估计。[Ghavamzadeh and Engel, 2006] 描述了解决该问题的两种思路: 用蒙特卡罗方法估计 G 和用最大似然法估计 MDP 动力学并计算 G 。他们用实际结果表明, 即使用上述方法估计 G , BPG 可以比基于蒙特卡罗的 PG 算法表现得更好。

可以通过稀疏化大幅度提升 BPG 在时间和空间的效率。该稀疏化过程能够增量式执行, 并且有助于在内核矩阵是奇异矩阵或者近乎奇异的情况下提高算法的数值稳定性。与 GPTD 类似, 一种可能是使用 [Engel et al, 2002] 提出的在线稀疏化方法为路径的一个字典 (dictionary) 集合选择性地增加一条新的观测路径, 其中字典集合可以用作对最终解进行近似的一个基。最后, 容易看出 BPG 模型和算法可以使用类似于 [Baxter and Bartlett, 2001] 提出的方法扩展到 POMDP。

11.2.3 演员 - 评论家算法

演员 - 评论家 (AC) 算法是最早研究的强化学习算法之一 [Barto et al, 1983; Sutton, 1984]。这些算法组成了强化学习算法的一个家族, 其思路是维护两类不同的算法成分: 演员和评论家。其中演员的作用是维护和更新动作选择策略, 而评论家的作用是估计演员策略的相应价值函数。常见的做法是演员使用随机梯度上升方法更新策略参数, 同时评论家使用某种形式的时序差分 (TD) 学习方法估计价值函数 [Sutton, 1988]。当用于演员和评论家的表征在 [Sutton et al, 2000] 以及 [Konda and Tsitsiklis, 2000] 阐述的意义下兼容时, 相应演员 - 评论家算法是一种简单而优雅的方法, 并且保证收敛 (在适当条件下) 到评论家使用的性能指标的局部最优解加上函数近似中固有的时序差分误差度量 [Konda and Tsitsiklis, 2000; Bhatnagar et al, 2009]。演员 - 评论家算法 (例如 [Sutton et al, 2000; Konda and Tsitsiklis, 2000; Peters et al, 2005; Bhatnagar et al, 2007]) 相对于没有评论家角色的策略梯度方法的显著优势在于, 评论家的引入减少了策略梯度估计的起伏, 使得策略空间的搜索更加有效并且鲁棒。

大多数演员 - 评论家算法建立在参数化评论的基础上, 该评论不断更新从而优化频率主

义的适应度准则。然而, 11.2.1 节描述的 GPTD 模型提供了一组贝叶斯主义的评论, 可以返回价值函数上的完整后验分布。本小节中, 我们研究在评论家部分整合了 GPTD 的贝叶斯演员-评论家算法 (BAC) [Ghavamzadeh and Engel, 2007]。我们将演示 GPTD 返回的后验矩怎样使得我们获得策略梯度后验矩的闭式。该路径之所以可行, 是由于利用 Fisher 内核 [Shawe-Taylor and Cristianini, 2004] 作为 GPTD 状态-动作优势值的先验协方差内核。这是 11.2.2 节讨论的 BPG 方法的自然延伸。需要重点注意的是, 虽然 BPG 中作为学习和推断基础的基本观测单元是一条完整的求解轨迹, BAC 利用系统轨迹的马尔可夫特性和单独的状态-动作-奖励迁移作为基本观测单元。这一特点有助于减少梯度估计的起伏, 导致相对于 BPG 和传统蒙特卡罗方法更加陡峭的学习曲线。

在一定的正则化条件下 [Sutton et al, 2000], 由公式 (11.12) 定义的策略 π 的期望回报可以写作:

$$\eta(\pi) = \int_{\mathcal{Z}} dz \mu^{\pi}(z) \bar{r}(z) \quad (369)$$

其中 $\bar{r}(z)$ 是状态-动作对 z 的平均奖励, 并且 $\mu^{\pi}(z) = \sum_{t=0}^{\infty} \gamma^t P_t^{\pi}(z)$ 是在遵循策略 π 遇到的状态-动作对的折扣权重。在 $\mu^{\pi}(z) = \mu^{\pi}(s, a)$ 中对 a 进行积分, 就得到在遵循策略 π 遇到的状态的折扣权重 $p^{\pi}(s) = \int_{\mathcal{A}} da \mu^{\pi}(s, a)$ 。不同于 p^{π} 和 μ^{π} , $(1-\gamma)p^{\pi}$ 和 $(1-\gamma)\mu^{\pi}$ 是概率分布。它们类似于无折扣情况下在状态和状态-动作对上的静态分布, 这是因为在 γ 趋于 1 时, 这些分布如果存在则趋于静态。策略梯度定理 [Marbach, 1998; Sutton et al, 2000; Konda and Tsitsiklis, 2000] 表明参数策略的期望回报梯度为:

$$\nabla \eta(\theta) = \int ds da \rho(s; \theta) \nabla \pi(a|s; \theta) Q(s, a; \theta) = \int dz \mu(z; \theta) \nabla \log \pi(a|s; \theta) Q(z; \theta) \quad (11.17)$$

可以观察到, 如果 $b: \mathcal{S} \rightarrow \mathbb{R}$ 是一个 s 的任意函数 (又称为基准线), 则有:

$$\begin{aligned} \int_{\mathcal{S}} ds da \rho(s; \theta) \nabla \pi(a|s; \theta) b(s) &= \int_{\mathcal{S}} ds \rho(s; \theta) b(s) \nabla \left(\int_{\mathcal{A}} da \pi(a|s; \theta) \right) \\ &= \int_{\mathcal{S}} ds \rho(s; \theta) b(s) \nabla(1) = 0 \end{aligned}$$

并且这样一来, 对任意基准线 $b(s)$, 公式 (11.17) 可以写作:

$$\nabla \eta(\theta) = \int_{\mathcal{Z}} dz \mu(z; \theta) \nabla \log \pi(a|s; \theta) [Q(z; \theta) + b(s)] \quad (11.18)$$

现在考虑针对给定策略 π 的动作价值函数的情况, 该函数 Q^{π} 由学到的近似函数表示。如果近似足够好, 则希望用其取代公式 (11.17) 和公式 (11.18) 中的 Q^{π} , 并且仍然能大致指向真实梯度的方向。[Sutton et al 2000]; [Konda and Tsitsiklis 2000] 发现, 如果以 w 为参数的近似函数 $\hat{Q}^{\pi}(\cdot; w)$ 具有兼容性, 即 $\nabla_w \hat{Q}^{\pi}(s, a; w) = \nabla \log \pi(a|s; w)$, 而且最小化以 w^* 为参数的均方误差:

$$\mathcal{E}^{\pi}(w) = \int_{\mathcal{Z}} dz \mu^{\pi}(z) [Q^{\pi}(z) - \hat{Q}^{\pi}(z; w)]^2 \quad (11.19)$$

则可以在公式 (11.17) 和公式 (11.18) 中用 $\hat{Q}^{\pi}(\cdot; w)$ 代替 Q^{π} 。基函数线性组合形式的近似动作价值函数可以写作 $\hat{Q}^{\pi}(z; w) = w^T \psi(z)$ 。如果 ψ 与策略兼容, 即 $\psi(z; w) = \nabla \log \pi(a|s; \theta)$, 则该近似具有兼容性。可以证明公式 (11.19) 的均方误差和下式

$$\mathcal{E}^{\pi}(w) = \int_{\mathcal{Z}} dz \mu^{\pi}(z) [Q^{\pi}(z) - w^T \psi(z) - b(s)]^2 \quad (11.20)$$

同解 (如 [Bhatnagar et al 2007, 2009]), 并且如果公式 (11.20) 中的参数 w 等于 w^* , 则结果的均方误差 $\mathcal{E}^{\pi}(w^*)$ 可以通过设置 $b(s) = V^{\pi}(s)$ 进一步减小 [Bhatnagar et al, 2007, 2009]。换言之, 如果基准线选为价值函数本身, 则能最小化动作价值函数估计的起伏。这意味着更

价值函数 $Q^\pi(s, a)$ 。

现在我们能够描述 BAC 方法背后的主要思想。使用公式 (11.17) Q 的线性, 并利用 $g(\mathbf{z}; \boldsymbol{\omega}) = \mu^\pi(\mathbf{z}) \nabla \log \pi(a|s; \boldsymbol{\theta})$, 我们可以为策略梯度的后验矩得到下式 [O'Hagan, 1991]:

$$\begin{aligned}\mathbb{E}[\nabla \eta(\boldsymbol{\theta})|\mathcal{Q}_i] &= \int_{\mathcal{Z}} d\mathbf{z} g(\mathbf{z}; \boldsymbol{\theta}) \hat{Q}_i(\mathbf{z}; \boldsymbol{\theta}) = \int_{\mathcal{Z}} d\mathbf{z} g(\mathbf{z}; \boldsymbol{\theta}) \mathbf{k}_i(\mathbf{z})^\top \boldsymbol{\alpha}_i \\ \text{Cov}[\nabla \eta(\boldsymbol{\theta})|\mathcal{Q}_i] &= \int_{\mathcal{Z}^2} d\mathbf{z} d\mathbf{z}' g(\mathbf{z}; \boldsymbol{\theta}) \hat{S}_i(\mathbf{z}, \mathbf{z}') g(\mathbf{z}'; \boldsymbol{\theta})^\top \\ &= \int_{\mathcal{Z}^2} d\mathbf{z} d\mathbf{z}' g(\mathbf{z}; \boldsymbol{\theta}) (k(\mathbf{z}, \mathbf{z}') - \mathbf{k}_i(\mathbf{z})^\top \mathbf{C}_i \mathbf{k}_i(\mathbf{z}')) g(\mathbf{z}'; \boldsymbol{\theta})^\top\end{aligned}\quad (11.21)$$

其中 \hat{Q}_i 和 \hat{S}_i 是通过 GPTD 评论家按照公式 (11.9) 计算的 Q 的后验矩。

这些公式提供了策略梯度后验矩的一般形式。现在我们还剩下一个计算问题, 即怎样计算前面已经反复出现的积分式:

$$\mathbf{U}_i = \int_{\mathcal{Z}} d\mathbf{z} g(\mathbf{z}; \boldsymbol{\theta}) \mathbf{k}_i(\mathbf{z})^\top \text{ 和 } \mathbf{V} = \int_{\mathcal{Z}^2} d\mathbf{z} d\mathbf{z}' g(\mathbf{z}; \boldsymbol{\theta}) k(\mathbf{z}, \mathbf{z}') g(\mathbf{z}'; \boldsymbol{\theta})^\top \quad (11.22)$$

使用公式 (11.22) 的定义, 我们可以把梯度后验矩简洁地写为:

$$\mathbb{E}[\nabla \eta(\boldsymbol{\theta})|\mathcal{Q}_i] = \mathbf{U}_i \boldsymbol{\alpha}_i \text{ 和 } \text{Cov}[\nabla \eta(\boldsymbol{\theta})|\mathcal{Q}_i] = \mathbf{V} - \mathbf{U}_i \mathbf{C}_i \mathbf{U}_i^\top \quad (11.23)$$

[Ghavamzadeh and Engel, 2007] 指出为了能够计算上面的积分式, 先验协方差内核应将该定义为 $k(\mathbf{z}, \mathbf{z}') = k_s(s, s') + k_F(\mathbf{z}, \mathbf{z}')$, 即任意状态内核 k_s 和状态对之间的 Fisher 内核 $k_F(\mathbf{z}, \mathbf{z}') = \mathbf{u}(\mathbf{z})^\top \mathbf{G}(\boldsymbol{\theta})^{-1} \mathbf{u}(\mathbf{z}')$ 之和。他们证明使用这样的先验协方差内核, 公式 (11.22) 的 \mathbf{U}_i 和 \mathbf{V} 满足 $\mathbf{U}_i = [\mathbf{u}(\mathbf{z}_0), \dots, \mathbf{u}(\mathbf{z}_i)]$ 和 $\mathbf{V} = \mathbf{G}(\boldsymbol{\theta})$ 。当可以获得期望回报的梯度的后验矩时, 贝叶斯演员-评论家算法可以通过在均值方向上更新策略参数而很容易地获得。

类似于 11.2.2 节 BPG 的情况, 每个策略的 Fisher 信息矩阵可以通过蒙特卡罗和最大似然方法进行估计, 并且可以经由在线稀疏化方法 [Engel et al, 2002], 使得该算法在时间和空间上异常高效并且局部数值更鲁棒。

11.3 基于模型的贝叶斯强化学习

在基于模型的贝叶斯强化学习中, 我们显式地估计在与系统交互时的环境动力学模型。我们从 MDP 模型未知参数的先验信念开始。接下来, 当观测到未知参数的表现形式时, 我们更新信念从而反映观测数据。在离散状态-动作 MDP 的情况下, 每个未知迁移概率 $P(s'|s, a)$ 都是一个取值在 $[0, 1]$ 范围的未知参数 $\theta_a^{s, s'}$, 因此信念就是在连续间隔定义的概率密度。基于模型的技术一般会比无模型技术具有更高的计算复杂度, 但是允许运行环境的先验知识更加自然地整合到学习过程中。

11.3.1 由 POMDP 表述的贝叶斯强化学习

基于模型的贝叶斯强化学习可以表述为一个部分可观测马尔可夫决策过程 (POMDP) [Duff, 2002], 即可以形式化的表示为 $\langle \mathcal{O}_p, \mathcal{A}_p, \mathcal{O}_p, T_p, Z_p, R_p \rangle$ 。这里, $\mathcal{O}_p = \mathcal{O}\{\theta_a^{s, s'}\}$ 是混合状态集合, 由以下两部分的叉积定义: (离散、完全可观测的) 正态 MDP 状态集合 s 与 (连续、不可观测的) 模型参数 $\theta_a^{s, s'}$ (其中每个可行的 MDP 状态-动作-状态迁移有一个参数)。POMDP 的动作空间 $\mathcal{A}_p = \mathcal{A}$ 与 MDP 相同。观测空间与 MDP 的状态空间正好吻合, 即 $\mathcal{O}_p = \mathcal{O}$, 这是由于 MDP 的状态空间完全可观测。迁移函数 $T_p(s, \theta, a, s', \theta') = P(s', \theta'|s, \theta, a)$ 可以分解为两个条件分布, 一个针对 MDP 状态, 即 $P(s'|s, \theta_a^{s, s'}, a) = \theta_a^{s, s'}$, 另一个针对未知参

数, 即 $P(\theta'|\theta) = \delta_{\theta}(\theta')$, 其中 $\delta_{\theta}(\theta')$ 是 Kronecker δ 函数 (当 $\theta' = \theta$ 时该函数值为 1, 否则为 0)。该 Kronecker δ 函数反映了未知参数均为静态的假设, 即 θ 不随时间改变。观测函数 $Z_p(s', \theta', a, o) = P(o|s', \theta', a)$ 指出在执行动作 a 到达联合状态 s', θ' 时看到观测 o 的概率。由于观测也是 MDP 状态, 因此 $P(o|s', \theta', a) = \delta_{s'}(o)$ 。

通过在未知参数 $\theta_a^{s,s'}$ 上定义信念, 可以在以上 POMDP 基础上表述一个信念-状态 MDP。这里关键点在于虽然原始强化学习问题牵涉到隐含变量, 信念-状态 MDP 是可观测的。该表述有效地把强化学习问题转化为在未知 MDP 模型参数上定义的信念空间的规划问题。

对于离散 MDP 来说, 一种自然的信念表征方式是 Dirichlet 分布, 这是由于 Dirichlet 分布是多项式 (multinomial) 的共轭密度 [DeGroot, 1970]。在多项式 p 上的 Dirichlet 分布 $\text{Dir}(p; n) \propto \prod_i p_i^{n_i-1}$ 可以通过一个正数 n_i 参数化, 从而使得 n_i-1 能够解释为概率为 p_i 的事件已经被观测到的次数。由于可行迁移 s, a, s' 只和未知参数有关, 我们可以把信念建模为 Dirichlets 积, 其中每个信念针对一个未知模型参数 $\theta_a^{s,s'}$ 。

在这个 POMDP 中的信念监测等同于基于观测状态迁移用贝叶斯规则更新信念。对定义在某个迁移参数 θ 上的先验信念 $b(\theta) = \text{Dir}(p; n)$ 来说, 当环境中观测到特定 (s, a, s') 迁移时, 后验信念通过贝叶斯规则计算, $b'(\theta) \propto \theta_a^{s,s'} b(\theta)$ 。如果用由当前状态 s 和针对每个 Dirichlet 分布的超参数 $n_a^{s,s'}$ 组成的二元组 $\langle s, \{n_a^{s,s'}\} \rangle$ 表示信念状态, 信念更新就变为把当前状态更新为 s' 并把超参数 $n_a^{s,s'}$ 加 1, 从而匹配观测状态迁移 s, a, s' 。

用 POMDP 表述的贝叶斯强化学习提供了自然的框架去推理探索/利用的折衷。由于信念编码了学习器获得的全部信息 (即对过去动作和观测的充分历史统计), 同时优化的 POMDP 策略是一个从信念到动作并最大化总奖励期望的映射, 因此优化的 POMDP 策略自然地优化探索/利用的折衷。换言之, 由于平衡利用 (即刻的增益) 和探索 (信息的增益) 的目标就是最大化总体奖励之和, 所以最佳 POMDP 策略导致最好的折衷。不过要注意的是, 以上假设了先验信念是准确的并且计算是精确的, 这当然在实践中并不常见。然而, POMDP 表述提供了一种有用的形式化方法, 从而可以设计自然折衷探索/利用的算法。

POMDP 表述把强化学习规约为一个拥有特殊结构的规划问题。在下一节我们推导最优价值函数的参数化, 该问题能够通过动态规划精确计算 [Poupart et al, 2006]。但是, 由于计算复杂度随着规划视野指数级增加, 我们也讨论近似方法。

11.3.2 通过动态规划的贝叶斯强化学习

利用贝叶斯强化学习的 POMDP 观测相当于常态 MDP 状态的事实, 信念-状态 MDP 的优化问题中针对最优价值函数的贝尔曼方程 [Duff, 2002] 为:

$$V_s^*(b) = \max_a R(s, a) + \gamma \sum_{s'} P(s'|s, b, a) V_{s'}^*(b_a^{s,s'}) \quad (11.24)$$

这里 s 是当前常态 MDP 状态, b 是定义在模型参数 θ 上的当前信念, $b_a^{s,s'}$ 是在迁移 s, a, s' 之后更新过的信念。迁移模型定义如下:

$$P(s'|s, b, a) = \int_{\theta} d\theta b(\theta) P(s'|s, \theta, a) = \int_{\theta} d\theta b(\theta) \theta_a^{s,s'} \quad (11.25) \quad [373]$$

上式就是针对信念 b 的平均迁移概率 $P(s'|s, a)$ 。由于最优 POMDP 策略定义为达到最高可达的未来期望奖励, 那么这样的策略应该自动优化原始强化学习问题中的探索/利用的折衷。

已知(例如参见第12章)对应于离散状态和动作空间 POMDP 的最优有限域价值函数是分段线性的凸函数,这相当于称为 α 向量的线段集合 Γ 的上界,即 $V^*(b) = \max_{\alpha \in \Gamma} \alpha(b)$ 。在文献中, α 被定义为 b 的线性函数(即 $\alpha(b)$) 以及 s 的向量(即 $\alpha(s)$), 满足以下条件 $\alpha(b) = \sum_s b(s) \alpha(s)$ 。因此,对于离散 POMDP, 价值函数可以被一组 α 向量参数化,其中每个向量表示为每个状态价值的向量。方便的是,该参数化过程在贝尔曼备份运算符下是封闭的。

在贝叶斯强化学习中,虽然状态空间是混合型的, [Duff, 2002]; [Porta et al, 2005] 的工作表明其价值函数的分段线性和凸性仍然可以成立。特别是,离散动作 POMDP 的最优有限域价值函数对应于称为 α 函数的线段集合 Γ 的上界(由于 POMDP 状态集合 θ 的连续本质),而 α 函数可以根据每个常态状态 s 组合为子集:

$$V_s^*(b) = \max_{\alpha \in \Gamma} \alpha_s(b) \quad (11.26)$$

这里 α 可以定义为一个以 s 为下标的 b 的线性函数(即 $\alpha_s(b)$) 或者是一个以 s 为下标的 θ 的函数(即 $\alpha_s(\theta)$) 并满足:

$$\alpha_s(b) = \int_{\theta} d\theta b(\theta) \alpha_s(\theta) \quad (11.27)$$

因此,贝叶斯强化学习的价值函数可以参数化为一组 α 函数。不仅如此,类似于 POMDP, α 函数可以像我们之后会介绍的那样通过动态规划进行更新。但是,贝叶斯强化学习中, α 函数的表征的复杂度随着动态规划备份数量而增长:对于域 T 来说,最优价值函数可能涉及 T 上的指数个 α 函数,而且每个 α 函数的表征也在 T 上具有指数级复杂度(例如,在基函数级数中非零系数个数)。

11.3.2.1 价值函数参数化

假定针对剩余 k 个步骤的最优价值函数 $V_s^k(b)$ 由 α 函数集合 Γ^k 组成,并且满足 $V_s^k(b) = \max_{\alpha \in \Gamma^k} \alpha_s(b)$ 。利用贝尔曼方程,我们可以通过动态规划计算剩余 $k+1$ 个步骤时的最优价值函数 $V_s^{k+1}(b)$ 的表征集合 Γ^{k+1} 。首先,我们重写公式(11.24),用 V^k 代替 Γ^k 中关于 α 函数的最大化:

$$V_s^{k+1}(b) = \max_a R(b, a) + \gamma \sum_{s'} P(s' | s, b, a) \max_{\alpha \in \Gamma^k} \alpha_{s'}(b^{s, s'})$$

接下来,我们可以用三个步骤分解贝尔曼方程。第一个步骤是找到关于 a 和 s' 的最大 α 函数,第二个步骤找到最佳动作 a ,第三个步骤使用最佳动作和最大 α 函数执行实际的贝尔曼备份:

$$\alpha_{b, a}^{s, s'} = \arg \max_{\alpha \in \Gamma^k} \alpha(b_{a, a}^{s, s'}) \quad (11.28)$$

$$a_b^s = \arg \max_a R(s, a) + \gamma \sum_{s'} P(s' | s, b, a) \alpha_{b, a}^{s, s'}(b_{a, a}^{s, s'}) \quad (11.29)$$

$$V_s^{k+1}(b) = R(s, a_b^s) + \gamma \sum_{s'} P(s' | s, b, a_b^s) \alpha_{b, a_b^s}^{s, s'}(b_{a_b^s, a_b^s}^{s, s'}) \quad (11.30)$$

我们进而可以使用由 θ 表示(而不是由 b) 的 α 函数重写第三个步骤并扩展信念状态 $b_{a_b^s}^{s, s'}$:

$$V_s^{k+1}(b) = R(s, a_b^s) + \gamma \sum_{s'} P(s' | s, b, a_b^s) \int_{\theta} d\theta b_{a_b^s}^{s, s'}(\theta) \alpha_{b, a_b^s}^{s, s'}(\theta) \quad (11.31)$$

$$= R(s, a_b^s) + \gamma \sum_{s'} P(s' | s, b, a_b^s) \int_{\theta} d\theta \frac{b(\theta) P(s' | s, \theta, a_b^s)}{P(s' | s, b, a_b^s)} \alpha_{b, a_b^s}^{s, s'}(\theta) \quad (11.32)$$

$$= R(s, a_b^s) + \gamma \sum_{s'} \int_{\theta} d\theta b(\theta) P(s' | s, \theta, a_b^s) \alpha_{b, a_b^s}^{s, s'}(\theta) \quad (11.33)$$

$$= \int_{\theta} d\theta b(\theta) \left[R(s, a_b^s) + \gamma \sum_{s'} P(s' | s, \theta, a_b^s) \alpha_{b, a_b^s}^{s, s'}(\theta) \right] \quad (11.34)$$

方括号中的表达式是 s 和 θ 的函数，因此我们可以用它来定义 Γ^{k+1} 中的 α 函数：

$$\alpha_{b,s}(\theta) = R(s, a_b^s) + \gamma \sum_{s'} P(s' | s, \theta, a_b^s) \alpha_{b,a_b^s}^{s,s'}(\theta) \quad (11.35)$$

我们针对每个 b 定义一个如上的 α 函数，所有 $\alpha_{b,s}$ 组成集合 Γ^{k+1} 。由于每个 $\alpha_{b,s}$ 是使用最优动作和 Γ^k 上的 α 函数而定义的，因此 $\alpha_{b,s}$ 在 b 满足必要优化，我们可以引入一个针对所有 α 函数的最大值而不损失精度：

$$V_s^{k+1}(b) = \int_{\theta} d\theta b(\theta) \alpha_{b,s}(\theta) = \alpha_s(b) = \max_{\alpha \in \Gamma^{k+1}} \alpha_s(b) \quad (11.36)$$

根据以上推导公式，我们有下述定理（证明请参阅原始文献）：

定理 11.1[Poupart et al, 2006]：贝叶斯强化学习的 α 函数是（非标准的）Dirichlets 乘积的线性组合。

375

注意上述定理中， α 函数表征的复杂度随动态规划备份的数量而增加：使用该定理和公式 (11.35)，可知每个 α 函数的项数随着每个备份以 $O(|\mathcal{S}|)$ 速度增长，这导致总项数以指数级速度随着规划域增加。为了减轻项数指数级增加的问题，我们可以把各项的线性组合投影到更少的项（例如一个单项式基）。[Poupart et al, 2006] 描述了若干针对此目的的投影方案。

11.3.2.2 精确和近似动态规划算法

在得到 α 函数在贝尔曼备份下具有封闭性的表征形式后，就可以把几种针对离散 POMDP 的算法移植到贝叶斯强化学习。例如，可以通过借助于一种类似 Monahan 枚举算法的 POMDP 求解技术计算最优有限域的贝叶斯强化学习控制器（参见第 12 章），但是在每个备份中支持 α 函数的数量一般来说是 $|\mathcal{S}|$ 的指数函数。

一种不同的思路是，我们可以设计近似（点近似）的值迭代算法，利用价值函数通过 α 函数的参数化。例如，[Poupart et al, 2006] 提出了针对贝叶斯强化学习的 BEETLE 算法，该算法是针对离散 POMDP 的 Perseus 算法的扩展 [Spaan and Vlassis, 2005]。在该算法中，通过仿真随机策略采样一个 (s, b) 对的可达集合，继而（近似）值迭代通过在 (s, b) 对采样执行针对 α 函数特殊参数化的、基于点的后备而实现。

在值迭代中使用 α 函数这一特点，允许设计离线（即预编译）求解工具，这是由于 α 函数参数化提供了向信念空间非采样区域的推广。BEETLE 是文献中唯一的已知算法，能够利用 α 函数的这种形式从而实现在基于模型的贝叶斯强化学习问题中推广。另一方面，也可以使用任意通用函数近似方法。例如，[Duff, 2003] 描述了使用 (s, θ) 上特征的线性组合来近似价值函数的演员 - 评论家算法。大多数其他基于模型的贝叶斯强化学习算法是在线求解器，无法显式地对价值函数参数化。我们接下来描述其中一些算法。

11.3.3 近似在线算法

在线算法尝试通过推理当前的信念而近似贝叶斯最优动作，经常导致短视的动作选择策略。该方法避免了离线规划的额外开销（类似于 BEETLE），但是可能需要在运行时考虑多方面问题，因而在实际应用中比较昂贵。

376

早期的近似在线强化学习算法都是基于置信区间 [Kaelbling, 1993 ; Meuleau and Bourguine, 1999 ; Wiering, 1999] 或者针对动作选择的完美信息价值 (VPI) 原则 [Dearden et al, 1999]，但是两种方法均导致短视的动作选择策略。后者涉及在当前信念的支持下针对 MDP 估计最优 Q 值的分布，并用此估计计算从一个动作切换到另一个（希望更好的）动作的“增益”。我们不构造 Q 值的显式分布（类似 11.2.1.1 节），而是利用 $P(\theta)$ 模型的分布

去采样并计算每个模型的最优 Q 值。这样产生的 Q 值采样能够近似基础的 Q 值分布。每个动作的探索增益继而通过公式 (11.2) 进行估计, 其中通过样本均值近似期望的 Q 值。类似于公式 (11.1), 完美信息价值可以近似为:

$$\text{VPI}(s, a) \approx \frac{1}{\sum_i w_\theta^i} \sum_i w_\theta^i \text{Gain}_{s,a}(q_{s,a}^i) \quad (11.37)$$

其中 w_θ^i 是依赖于建议分布的采样模型的重要性权重。[Dearden et al, 1999] 描述了几种高效的程序从而根据建议分布对模型进行采样, 这些建议分布比 $P(\theta)$ 更容易处理。

另一种短视的贝叶斯动作选择策略是 Thompson 采样, 其中包含从当前信念中仅采样一个 MDP, 求解其最优 MDP (例如通过动态规划) 并在当前状态执行最优动作 [Thompson, 1933; Strens, 2000]。有报告指出该方法易于导致过度探索 [Wang et al, 2005]。

也可以通过在 POMDP (参阅前一节) 的信念-状态 MDP 计算接近最优策略实现一种不太短视的动作选择策略。由于这只有一个 MDP (虽然具有连续性并有特殊结构), 可以使用任何 MDP 的近似求解器。[Wang et al (2005); Ross and Pineau (2008)] 深入研究了 this 思想, 在信念-状态 MDP 上使用 [Kearns et al, 1999] 提出的稀疏采样算法。该方法使用一种显式的“向前看”方法检查起始于现有信念的有效域, 并通过动态规划和线性规划 [Castro and Precup, 2007] 的树形结构备份奖励, 从而形成接近贝叶斯最优的探索动作。这种基于树的搜索必须在每个时间步上进行, 导致实际应用中过于昂贵。有可能稀疏采样技术可以结合通过类似于 BEETLE 的 α 函数参数化而推广到整个信念空间的方法, 但是目前尚未看到此类算法发表。

11.3.4 贝叶斯多任务强化学习

多任务学习 (Multi-Task Learning, MTL) 是重要的学习范式, 近来已经成为机器学习的活跃研究领域 (例如 [Caruana, 1997; Baxter, 2000])。通常的设置是这样的, 即关注改进多个相关任务的学习, 由于可以在任务间共享信息, 因此其性能应该超过单一任务的学习。在针对每一个任务只有有限数据时, 以上信息的迁移及其重要。利用相关问题的数据提供了更多的训练样本, 从而改进最终解的性能。更正式的说来, MTL 的主要目标是最大化相对于单个学习任务平均性能的总体改进程度。注意区分该方法与迁移学习, 后者的目标是从一类任务中学习一个适当的偏置从而最大化期望未来性能。

大多数强化学习算法经常需要大量样本才能解决问题, 而不能直接利用来自其他相似任务的信息。但是, 近来的研究工作表明强化学习可以利用迁移和多任务学习技术减少达到接近最优解所需要的样本数量。所有多任务强化学习 (MTRL) 方法都假定任务在问题的某些部分享有相似性, 这些部分可以是动力学、奖励结构或价值函数。一些方法显式地假定共享成分可以从共同的生成式模型产生 [Wilson et al, 2007; Mehta et al, 2008; Lazaric and Ghavamzadeh, 2010], 但是该假设在其他一些工作中是更加隐含的 [Taylor et al, 2007; Lazaric et al, 2008]。在 [Mehta et al, 2008] 中, 若干任务共享同样的动力学和奖励特征, 而只在奖励函数的权重上有所区别。他们提出的方法使用从前面任务学到的价值函数作为先验来初始化新任务的价值函数。[Wilson et al, 2007; Lazaric and Ghavamzadeh, 2010] 都假定任务的某些部分的分布可以从一个层次贝叶斯模型 (Hierarchical Bayesian Model, HBM) 产生。我们将在下面给出以上两种方法的更多细节。

[Lazaric and Ghavamzadeh, 2010] 研究了 MTRL 场景, 其中学习器接收到一些拥有共

同状态和动作空间的 MDP。对任何给定策略，每个 MDP 只能产生少量样本，其数量不足以支撑对策略进行准确评估。在这一 MTRL 问题中，有必要确定拥有相同结构的任务类别并对其进行联合学习。值得注意的是一个任务是一个 MDP 和策略对，其中所有 MDP 拥有同样的状态和动作空间。Lazaric 和 Ghavamzadeh 考虑一种特定类型的 MTRL 问题，其中的任务共享价值函数结构。为了使得价值函数能够拥有共同的结构，我们假定它们都来自于同一先验分布的采样。两位作者对每一任务采用 GPTD 价值函数（参见 11.2.1 节），使用 HBM 对价值函数的分布建模，并且提出了针对以下问题的解法：1）价值函数联合学习（多任务学习）；2）有效迁移从前一种情况中获取的信息从而帮助新观察到的任务学习价值函数（迁移学习）。他们首次提出为所有价值函数属于一类的问题使用 HBM，并且推导了相应的 EM 算法去找到价值函数和模型超参数的 MAP 估计。但是，如果函数不属于同一类型，简单的联合学习甚至可能有害（负面迁移）。因此，重要的是建立能够从相关任务普遍受益、并且在不相关任务上不损害性能模型。这对强化学习尤其重要，因为在每个策略迭代步骤上（对拟合的值迭代同样成立）改变策略可以改变任务聚类的方式。以上情况意味着即使我们开始于完全属于同一类的价值函数，在一次迭代后新的价值函数可能被聚类到若干不同类别。为了解决这个问题，对于价值函数属于若干类的情况，他们引入了基于 HBM 的狄利克雷过程（DP），并且在这个模型中推导多任务和迁移学习场景的推理算法。

378

[Wilson et al, 2007] 中的 MTRL 方法也使用了基于动态规划的 HBM 对任务共同结构的分布进行建模。该工作中，任务共享动力学和奖励函数的结构。其配置是增量式的，即任务以序列形式观测，每个任务产生的样本数量则不作约束。此处的焦点不在于在有限样本上联合学习，而在于使用前面获得的信息帮助新的任务学习。换言之，该工作侧重于迁移学习而不是多任务学习。

11.3.5 集成先验知识

当迁移学习和多任务学习不可用的时候，学习器可以仍然希望使用领域知识从而减少学习任务的复杂度。在非贝叶斯强化学习中，领域知识通常隐式地编码于编码状态空间的特征选择，或者价值函数的参数形式，再或者所考虑的策略类型。在贝叶斯强化学习中，先验分布提供了一种显式的并且表达力很强的机制，从而能够编码领域知识。与其开始于一个不具信息量的先验（例如均匀分布或 Jeffrey 先验），不如通过指定先验分布将学习过程偏置到领域专家感觉更可能的方向，从而直接减少数据需求。

例如，在基于模型的贝叶斯强化学习中，在迁移和奖励上的 Dirichlet 分布可以自然编码专家偏置。回想一下，Dirichlet 超参数 $n_i - 1$ 可以解释为以 p_i 为概率的事件被观测到的次数。因此，如果专家能够获取先验数据，其中事件发生了 $n_i - 1$ 次，或者推测在假想实验中事件会发生 $n_i - 1$ 次，则相应的 Dirichlet 分布可以用作“有信息量”的先验分布。另一条途径是，如果拥有某种信念或者先验数据可以估计某个未知多项分布的均值和方差，则 Dirichlet 的超参数可以通过矩匹配方法设置。

Dirichlet 分布的确定是其只允许表示单模型先验分布。然而，混合 Dirichlet 分布可用于表示多项分布。实际上，由于 Dirichlet 具有多项式形式（即 $\text{Dir}(\theta) = \prod_i \theta_i^{n_i}$ ），所以混合 Dirichlet 是拥有正系数的多项式（即 $\sum_j c_j \prod_i \theta_i^{n_{ij}}$ ）。因此，如果拥有足够大数量的混合成分，就可以任意逼近任何想要的未知多项先验分布。[Pavlov and Poupart, 2008] 探索了使用混合 Dirichlet 表示模型动力学和策略的联合先验。尽管混合 Dirichlet 表示能力很强，在某些情况

379

下还可以用生成式模型构造先验。为了达到这一效果, [Doshi-Velez et al, 2010] 研究使用诸如层次 Dirichlet 过程的层次先验, 其中模型动力学和策略表示为随机有限状态控制器。前一节描述的多任务和迁移学习也探索了针对价值函数 [Lazaric and Ghavamzadeh, 2010] 和系统动力学 [Wilson et al, 2007] 的层次先验。

11.4 有限样本分析和复杂度问题

贝叶斯强化学习最吸引人之处在于, 提供了一种为给定策略获得有限样本估计的方法, 该估计以价值和起伏的后验分布形式表征。该思想首先由 [Mannor et al, 2007] 提出, 研究了某一策略的价值函数估计的偏置和方差。假定有一个外生的采样过程 (即只能观测到迁移和奖励, 但不能控制), 存在一个常态模型 (例如通过最大后验概率估计获得) 和一个定义在所有可能模型上的后验概率分布。给定策略 π 和模型 $\theta = \langle T, r \rangle$ 上的后验分布, 我们可以考虑如下的期望后验价值函数:

$$\mathbb{E}_{\tilde{T}, \tilde{r}} \left[\mathbb{E}_s \left[\sum_{t=1}^{\infty} \gamma^t \tilde{r}(s_t) \middle| \tilde{T} \right] \right] \quad (11.38)$$

其中外层期望是根据 MDP 模型参数的后验而来, 而内层期望是针对给定模型参数下的迁移。处理无穷和后, 我们有:

$$\mathbb{E}_{\tilde{T}, \tilde{r}} \left[\left(I - \gamma \tilde{T}_{\pi} \right)^{-1} \tilde{r}_{\pi} \right] \quad (11.39)$$

其中 \tilde{T}_{π} 和 \tilde{r}_{π} 分别是策略 π 在 $\langle \tilde{T}, \tilde{r} \rangle$ 为真实模型时的迁移矩阵和奖励向量。该问题最大化定义在轨迹和模型随机变量上的期望回报。由于在期望回报上 \tilde{T} 的非线性效应, [Mannor et al, 2007] 认为对给定策略评估该问题的目标已经非常困难。

假定一个针对迁移的 Dirichlet 先验和一个针对奖励的高斯先验, 可以为给定策略的价值函数获得偏置和方差的估计。这些估计建立在公式 (11.39) 的一阶或二阶近似上。从计算角度看, 这些估计的处理相对容易, 而且价值函数可以去偏置。当尝试在策略空间优化时, [Mannor et al, 2007] 用实验表明常用的依赖于最有可能 (或期望) 参数的方法会导致在结果策略性能估计上的强偏置。

从贝叶斯视角看有限样本, 就很容易导向策略优化的问题, 即在公式 (11.38) 对所有策略额外计算一次最大值。在马尔可夫决策过程中的标准做法是考虑所谓的鲁棒方法: 假定问题参数属于某个不确定集合, 找到拥有最佳最坏情况性能的策略。这可以有效地通过动态规划类算法实现 (请参阅 [Nilim and El Ghaoui, 2005 ; Iyengar, 2005])。鲁棒方法的问题在于会导致过于保守的解。同时, 现有算法要求不同状态的不确定性互不相关, 意味着不确定集合就是各个状态不确定集合的笛卡尔积。

贝叶斯视角的好处之一是允许已知风险的方法, 这是由于我们拥有已有模型的概率分布。例如, 可以在此背景下考虑偏置 - 方差的折衷, 其中我们可以在方差约束下或者对过大方差进行惩罚的条件下最大化奖励。贝叶斯背景下的均值 - 方差优化看似比较困难, 而且目前还没有已知复杂性结果。为了减轻该问题的影响, [Delage and Mannor, 2010] 提出了针对一种风险 - 敏感百分比优化准则的近似:

$$\begin{aligned} & \text{maximize}_{y \in \mathbb{R}, \pi \in \mathcal{Y}} && y \\ & \text{s.t. } P_{\theta} \left(\mathbb{E}_s \left(\sum_{t=0}^{\infty} \gamma^t r_t(s_t) \middle| s_0 \propto q, \pi \right) \geq y \right) \geq 1 - \varepsilon \end{aligned} \quad (11.40)$$

对于给定策略 π , 上述机会约束问题给我们一个 $1-\varepsilon$ 的保证, 即 π 会比计算出的 y 表现出色。公式 (11.40) 中参数 ε 衡量出比 y 更差的策略风险。我们使用的性能瓶颈与风险-敏感准则有关, 这些准则经常用于金融业 (例如风险价值)。公式 (11.40) 定义的规划问题不像鲁棒方法 (实际上就是 $\varepsilon=0$ 的特例) 那样保守, 但在常态参数下仍不是乐观方法。从计算角度看, [Delage and Mannor, 2010] 证明该优化问题在一般条件下为 NP 难问题, 但是如果奖励后验为高斯后验并且迁移中不存在不确定性时多项式时间可解。不过, 如果奖励函数拥有高斯先验并且迁移函数拥有 Dirichlet 先验, 则二阶近似产生一个一般情况下可以计算的问题。

上述工作解决了在外生状态采样程序下策略优化和评估的问题, 下一个有趣的问题是从样本复杂度的角度考虑强化学习的探索-利用问题。虽然通过考虑定义在所有模型的分布并最大化期望奖励, 基于模型的贝叶斯强化学习为该问题引入了一个优雅的解决方案, 但是即使在最简单的问题上都存在计算量过大的问题 (请参阅 [Dimitrakakis, 2010] 关于随机搜索树近似的方法)。两篇最近的论文探讨基于模型的贝叶斯强化学习的复杂度问题。第一篇论文中, [Kolter and Ng, 2009] 提出了一个简单的算法, 并且证明该算法在多项式时间步骤后以高概率趋近于真正优化的 (计算量过大而不可行) 贝叶斯策略。该算法及其分析方法与 PAC-MDP (例如 [Brafman and Tennenholtz, 2002; Strehl et al, 2006]) 具有相似性, 但是其探索过程比 PAC-MDP 更贪婪。在第二篇文章中, [Asmuth et al, 2009] 提出一种驱动探索的方法, 即从后验中采样多个模型并乐观地选择动作。何时进行重采样和怎样组合模型的决策建立在乐观式启发之上。相关算法能够以较高概率达到接近最优的奖励, 并且相对于后验分布在学习收敛的速度而言, 采样复杂度较低。最后, [Fard and Pineau, 2010] 提出了一种 PAC-贝叶斯风格的上界, 允许在无分布的 PAC 和数据效率高的贝叶斯范式之间进行平衡。

381

11.5 总结和讨论

虽然贝叶斯强化学习也许是最早的一类强化学习方法, 早在 1960 年代就被运筹学学者进行了研究, 但是近来机器学习学者对其新一轮的兴趣仍然导致许多本章介绍的进展。其中大多数研究兴趣来自能够对关注的对象引入显式的分布。特别是, 一旦使用分布对模型不同部分的不确定性、价值函数和梯度进行量化, 则可以自动优化探索/利用的折衷。优化策略时也可以引入风险量。

本章对当前贝叶斯技术在单主体、完全可观测域强化学习问题的应用进行综述。我们注意到贝叶斯技术已经被用在部分可观察域 [Ross et al, 2007, 2008; Poupart and Vlassis, 2008; Doshi-Velez, 2009; Veness et al, 2010] 和多主体系统 [Chalkiadakis and Boutilier, 2003, 2004; Gmytrasiewicz and Doshi, 2005]。

参考文献

- Aharony, N., Zehavi, T., Engel, Y.: Learning wireless network association control with Gaussian process temporal difference methods. In: Proceedings of OPNETWORK (2005)
- Asmuth, J., Li, L., Littman, M.L., Nouri, A., Wingate, D.: A Bayesian sampling approach to exploration in reinforcement learning. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2009, pp. 19–26. AUAI Press (2009)
- Bagnell, J., Schneider, J.: Covariant policy search. In: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (2003)
- Barto, A., Sutton, R., Anderson, C.: Neuron-like elements that can solve difficult learning control problems. IEEE Transaction on Systems, Man and Cybernetics 13, 835–846 (1983)

382

- Baxter, J.: A model of inductive bias learning. *Journal of Artificial Intelligence Research* 12, 149–198 (2000)
- Baxter, J., Bartlett, P.: Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research* 15, 319–350 (2001)
- Bellman, R.: A problem in sequential design of experiments. *Sankhya* 16, 221–229 (1956)
- Bellman, R.: *Adaptive Control Processes: A Guided Tour*. Princeton University Press (1961)
- Bellman, R., Kalaba, R.: On adaptive control processes. *Transactions on Automatic Control*, IRE 4(2), 1–9 (1959)
- Bhatnagar, S., Sutton, R., Ghavamzadeh, M., Lee, M.: Incremental natural actor-critic algorithms. In: *Proceedings of Advances in Neural Information Processing Systems*, vol. 20, pp. 105–112. MIT Press (2007)
- Bhatnagar, S., Sutton, R., Ghavamzadeh, M., Lee, M.: Natural actor-critic algorithms. *Automatica* 45(11), 2471–2482 (2009)
- Brafman, R., Tennenholtz, M.: R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *JMLR* 3, 213–231 (2002)
- Caruana, R.: Multitask learning. *Machine Learning* 28(1), 41–75 (1997)
- Castro, P., Precup, D.: Using linear programming for Bayesian exploration in Markov decision processes. In: *Proc. 20th International Joint Conference on Artificial Intelligence* (2007)
- Chalkiadakis, G., Boutilier, C.: Coordination in multi-agent reinforcement learning: A Bayesian approach. In: *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 709–716 (2003)
- Chalkiadakis, G., Boutilier, C.: Bayesian reinforcement learning for coalition formation under uncertainty. In: *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 1090–1097 (2004)
- Cozzolino, J., Gonzales-Zubieta, R., Miller, R.L.: Markovian decision processes with uncertain transition probabilities. Tech. Rep. Technical Report No. 11, Research in the Control of Complex Systems. Operations Research Center, Massachusetts Institute of Technology (1965)
- Cozzolino, J.M.: Optimal sequential decision making under uncertainty. Master's thesis, Massachusetts Institute of Technology (1964)
- Dearden, R., Friedman, N., Russell, S.: Bayesian Q-learning. In: *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pp. 761–768 (1998)
- Dearden, R., Friedman, N., Andre, D.: Model based Bayesian exploration. In: *UAI*, pp. 150–159 (1999)
- DeGroot, M.H.: *Optimal Statistical Decisions*. McGraw-Hill, New York (1970)
- Delage, E., Mannor, S.: Percentile optimization for Markov decision processes with parameter uncertainty. *Operations Research* 58(1), 203–213 (2010)
- Dimitrakakis, C.: Complexity of stochastic branch and bound methods for belief tree search in bayesian reinforcement learning. In: *ICAART* (1), pp. 259–264 (2010)
- Doshi-Velez, F.: The infinite partially observable Markov decision process. In: *Neural Information Processing Systems* (2009)
- Doshi-Velez, F., Wingate, D., Roy, N., Tenenbaum, J.: Nonparametric Bayesian policy priors for reinforcement learning. In: *NIPS* (2010)
- Duff, M.: Optimal learning: Computational procedures for Bayes-adaptive Markov decision processes. PhD thesis, University of Massachusetts Amherst (2002)
- Duff, M.: Design for an optimal probe. In: *ICML*, pp. 131–138 (2003)
- Engel, Y.: Algorithms and representations for reinforcement learning. PhD thesis, The Hebrew University of Jerusalem, Israel (2005)
- Engel, Y., Mannor, S., Meir, R.: Sparse Online Greedy Support Vector Regression. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) *ECML 2002. LNCS (LNAI)*, vol. 2430, pp. 84–96. Springer, Heidelberg (2002)
- Engel, Y., Mannor, S., Meir, R.: Bayes meets Bellman: The Gaussian process approach to temporal difference learning. In: *Proceedings of the Twentieth International Conference on Machine Learning*, pp. 154–161 (2003)
- Engel, Y., Mannor, S., Meir, R.: Reinforcement learning with Gaussian processes. In: *Pro-*

- ceedings of the Twenty Second International Conference on Machine Learning, pp. 201–208 (2005a)
- Engel, Y., Szabo, P., Volkinshtein, D.: Learning to control an octopus arm with Gaussian process temporal difference methods. In: Proceedings of Advances in Neural Information Processing Systems, vol. 18, pp. 347–354. MIT Press (2005b)
- Fard, M.M., Pineau, J.: PAC-Bayesian model selection for reinforcement learning. In: Lafferty, J., Williams, C.K.I., Shawe-Taylor, J., Zemel, R., Culotta, A. (eds.) Advances in Neural Information Processing Systems, vol. 23, pp. 1624–1632 (2010)
- Ghavamzadeh, M., Engel, Y.: Bayesian policy gradient algorithms. In: Proceedings of Advances in Neural Information Processing Systems, vol. 19, MIT Press (2006)
- Ghavamzadeh, M., Engel, Y.: Bayesian Actor-Critic algorithms. In: Proceedings of the Twenty-Fourth International Conference on Machine Learning (2007)
- Gmytrasiewicz, P., Doshi, P.: A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research (JAIR)* 24, 49–79 (2005)
- Greensmith, E., Bartlett, P., Baxter, J.: Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research* 5, 1471–1530 (2004)
- Iyengar, G.N.: Robust dynamic programming. *Mathematics of Operations Research* 30(2), 257–280 (2005)
- Jaakkola, T., Haussler, D.: Exploiting generative models in discriminative classifiers. In: Proceedings of Advances in Neural Information Processing Systems, vol. 11, MIT Press (1999)
- Kaelbling, L.P.: Learning in Embedded Systems. MIT Press (1993)
- Kakade, S.: A natural policy gradient. In: Proceedings of Advances in Neural Information Processing Systems, vol. 14 (2002)
- Kearns, M., Mansour, Y., Ng, A.: A sparse sampling algorithm for near-optimal planning in large Markov decision processes. In: Proc. IJCAI (1999)
- Kolter, J.Z., Ng, A.Y.: Near-bayesian exploration in polynomial time. In: Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, pp. 513–520. ACM, New York (2009)
- Konda, V., Tsitsiklis, J.: Actor-Critic algorithms. In: Proceedings of Advances in Neural Information Processing Systems, vol. 12, pp. 1008–1014 (2000)
- Lazaric, A., Ghavamzadeh, M.: Bayesian multi-task reinforcement learning. In: Proceedings of the Twenty-Seventh International Conference on Machine Learning, pp. 599–606 (2010)
- Lazaric, A., Restelli, M., Bonarini, A.: Transfer of samples in batch reinforcement learning. In: Proceedings of ICML, vol. 25, pp. 544–551 (2008)
- Mannor, S., Simester, D., Sun, P., Tsitsiklis, J.N.: Bias and variance approximation in value function estimates. *Management Science* 53(2), 308–322 (2007)
- Marbach, P.: Simulated-based methods for Markov decision processes. PhD thesis, Massachusetts Institute of Technology (1998)
- Martin, J.J.: Bayesian decision problems and Markov chains. John Wiley, New York (1967)
- Mehta, N., Natarajan, S., Tadepalli, P., Fern, A.: Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning* 73(3), 289–312 (2008)
- Meuleau, N., Bourguin, P.: Exploration of multi-state environments: local measures and back-propagation of uncertainty. *Machine Learning* 35, 117–154 (1999)
- Nilim, A., El Ghaoui, L.: Robust control of Markov decision processes with uncertain transition matrices. *Operations Research* 53(5), 780–798 (2005)
- O’Hagan, A.: Monte Carlo is fundamentally unsound. *The Statistician* 36, 247–249 (1987)
- O’Hagan, A.: Bayes-Hermite quadrature. *Journal of Statistical Planning and Inference* 29, 245–260 (1991)
- Pavlov, M., Poupart, P.: Towards global reinforcement learning. In: NIPS Workshop on Model Uncertainty and Risk in Reinforcement Learning (2008)
- Peters, J., Schaal, S.: Reinforcement learning of motor skills with policy gradients. *Neural Networks* 21(4), 682–697 (2008)
- Peters, J., Vijayakumar, S., Schaal, S.: Reinforcement learning for humanoid robotics. In: Proceedings of the Third IEEE-RAS International Conference on Humanoid Robots (2003)

- Peters, J., Vijayakumar, S., Schaal, S.: Natural Actor-Critic. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 280–291. Springer, Heidelberg (2005)
- Porta, J.M., Spaan, M.T., Vlassis, N.: Robot planning in partially observable continuous domains. In: Proc. Robotics: Science and Systems (2005)
- Poupart, P., Vlassis, N.: Model-based Bayesian reinforcement learning in partially observable domains. In: International Symposium on Artificial Intelligence and Mathematics, ISAIM (2008)
- Poupart, P., Vlassis, N., Hoey, J., Regan, K.: An analytic solution to discrete Bayesian reinforcement learning. In: Proc. Int. Conf. on Machine Learning, Pittsburgh, USA (2006)
- Rasmussen, C., Ghahramani, Z.: Bayesian Monte Carlo. In: Proceedings of Advances in Neural Information Processing Systems, vol. 15, pp. 489–496. MIT Press (2003)
- Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. MIT Press (2006)
- Reisinger, J., Stone, P., Miikkulainen, R.: Online kernel selection for Bayesian reinforcement learning. In: Proceedings of the Twenty-Fifth Conference on Machine Learning, pp. 816–823 (2008)
- Ross, S., Pineau, J.: Model-based Bayesian reinforcement learning in large structured domains. In: Uncertainty in Artificial Intelligence, UAI (2008)
- Ross, S., Chaib-Draa, B., Pineau, J.: Bayes-adaptive POMDPs. In: Advances in Neural Information Processing Systems, NIPS (2007)
- Ross, S., Chaib-Draa, B., Pineau, J.: Bayesian reinforcement learning in continuous POMDPs with application to robot navigation. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2845–2851 (2008)
- Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press (2004)
- Silver, E.A.: Markov decision processes with uncertain transition probabilities or rewards. Tech. Rep. Technical Report No. 1, Research in the Control of Complex Systems. Operations Research Center, Massachusetts Institute of Technology (1963)
- Spaan, M.T.J., Vlassis, N.: Perseus: Randomized point-based value iteration for POMDPs. Journal of Artificial Intelligence Research 24, 195–220 (2005)
- Strehl, A.L., Li, L., Littman, M.L.: Incremental model-based learners with formal learning-time guarantees. In: UAI (2006)
- Strens, M.: A Bayesian framework for reinforcement learning. In: ICML (2000)
- Sutton, R.: Temporal credit assignment in reinforcement learning. PhD thesis, University of Massachusetts Amherst (1984)
- Sutton, R.: Learning to predict by the methods of temporal differences. Machine Learning 3, 9–44 (1988)
- Sutton, R., McAllester, D., Singh, S., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: Proceedings of Advances in Neural Information Processing Systems, vol. 12, pp. 1057–1063 (2000)
- Taylor, M., Stone, P., Liu, Y.: Transfer learning via inter-task mappings for temporal difference learning. JMLR 8, 2125–2167 (2007)
- Thompson, W.R.: On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika 25, 285–294 (1933)
- Veness, J., Ng, K.S., Hutter, M., Silver, D.: Reinforcement learning via AIXI approximation. In: AAAI (2010)
- Wang, T., Lizotte, D., Bowling, M., Schuurmans, D.: Bayesian sparse sampling for on-line reward optimization. In: ICML (2005)
- Watkins, C.: Learning from delayed rewards. PhD thesis, Kings College, Cambridge, England (1989)
- Wiering, M.: Explorations in efficient reinforcement learning. PhD thesis, University of Amsterdam (1999)
- Williams, R.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning 8, 229–256 (1992)
- Wilson, A., Fern, A., Ray, S., Tadepalli, P.: Multi-task reinforcement learning: A hierarchical Bayesian approach. In: Proceedings of ICML, vol. 24, pp. 1015–1022 (2007)

部分可观察的马尔可夫决策过程

Matthijs T.J. Spaan

摘要

对于学习器可以访问可靠状态信号的环境中的强化学习，基于马尔可夫决策过程 (MDP) 的方法取得了诸多成功。然而，在许多问题领域中，学习器受到有限的感知能力的阻碍，从而使其不能从其感知中恢复马尔可夫状态信号。如果考虑扩展 MDP 框架，部分可观察的马尔可夫决策过程 (POMDP) 就能允许在不确定的感测条件下进行原则性决策。在本章中，我们将通过关注那些区别于完全可观察的 MDP 模型的不同之处来呈现 POMDP 模型，并且展示了如何表示 POMDP 的最优策略。然后，我们会对基于模型的策略计算技术进行回顾，再概述 POMDP 的无模式方法。最后我们会总结并强调 POMDP 强化学习的最新趋势。

12.1 简介

已证明马尔可夫决策过程模型成功应用于学习如何在随机环境中行动。在本章中，我们通过松弛 MDP 模型的一个限制因素，即假设学习器充分确定环境状况来探索强化学习的方法。否则，学习器的传感器可以让它在任何时候完美地监视状态，即捕获的状态包含与最佳决策相关的环境的所有方面的信息。显然，这是一个强有力的假设，可以限制 MDP 框架的适用性。例如，当某些状态特征对学习器隐藏时，状态信号将不再是无后效过程的 (Markovian)，而这将违反大多数强化学习技术的关键假设 [Sutton and Barto, 1998]^①。

[387]

当将强化学习应用于具体学习器时，就会出现一个特别的关注。在许多机器人应用中，机器人的机载传感器不允许它明确地识别自己的位置或姿势 [Thrun et al, 2005]。此外，机器人的传感器通常被限制在观察其直接环境，并且可能不足以监视超出其附近区域的环境状态，即所谓的隐藏状态)。关于系统真实状态的另一个不确定因素来源是机器人传感器的缺陷。例如，让我们假设机器人使用相机来识别它正在交互的人物。处理相机图像的脸部识别算法有时会产生错误，并报告错误的身份。这种不完美的传感器也阻碍了机器人获知系统的真实状态：即使视觉算法报告 A，但仍有可能是 B 正与机器人交互。尽管在某些领域中，由于传感自身功能的不完善而导致的问题是可以被忽略的，但一般来说，这种不完善可能会导致严重的性能下降 [Singh et al, 1994]。

相反，在本章中，我们考虑扩展 (完全可观察的) MDP 设置，使其也能处理由学习器的不完美传感器导致的不确定性。一个部分可观察的马尔可夫决策过程 (POMDP) 允许学习器能在对其部分可观察到的环境中进行决策最优化 [Kaelbling et al, 1998]，这一点与 MDP 模型规定的完全可观察性不同。一般来说，部分可观察性来源于：1) 多个状态给出相同的

① 这个观点在书的前一部分很有可能已经提到了，出于一致性的考虑，这个引用可以使用之前正确的来代替。——编辑注

传感器读数,以防学习器只能感测环境的有限部分;2)其传感器读数混杂着噪声——观察相同的状态导致不同的传感器读数。部分可观察性可能导致“感知混叠”:对学习器的传感系统而言,环境的不同部分看起来很相似,但需要不同的动作。POMDP 在概率观察模型中捕获部分可观察性,其中该概率模型将可能的观察与状态相关联。

经典的 POMDP 示例是机器维护 [Smallwood and Sondik, 1973] 或结构检查 [Ellis et al, 1995] 的问题。在这些类型的问题中,学习器必须选择何时检查某个机器部件或桥接部分,以决定是否需要维护。但是,为了检查必须停止机器,或者要闭合桥接部分,这明显会带来经济成本。POMDP 模型可以适当平衡随时间变化的预期内的学习器劣化与调度检查或维护活动这两者之间的权衡。此外, POMDP 可以对这样的场景进行建模:系统仅选择对机器或桥接的状态信息进行审查,导致了一些缺陷不能总是可靠地被检查出来。最近, POMDP 模型多用于机器人应用,例如机器人导航 [Simmons and Koenig, 1995; Spaan and Vlassis, 2004; Roy et al, 2005; Foka and Trahanias, 2007]、主动感测 [Hoey and Little, 2007; Spaan et al, 2010]、对象抓握 [Hsiao et al, 2007] 或人机交互 [Doshi and Roy, 2008]。最后, POMDP 已用于不同领域,如医学治疗计划 [Hauskrecht and Fraser, 2000]、口语对话系统 [Williams and Young, 2007]、开发导航辅助 [Stankiewicz et al, 2007] 或入侵物种管理 [Haight and Polasky, 2010]。

本章的其余部分组织如下。首先,在 12.2 节中,我们正式介绍了 POMDP 模型,表明部分可观察性导致学习器对记忆或内部状态的需求。我们将讨论如何在 POMDP 框架中表示最优策略和价值函数。接下来,12.3 节回顾了 POMDP 的基于模型的技术,涉及最优、近似和启发式技术等。12.4 节概述了为 POMDP 开发或可应用于 POMDP 的无模型强化学习技术。最后,12.5 节介绍了 POMDP 强化学习的最新发展。

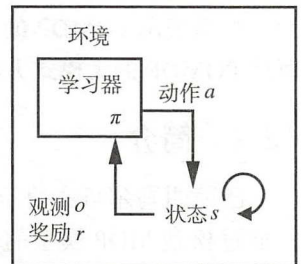


图 12.1 一个 POMDP 学习器与它所处的环境之间的交互

12.2 部分可观察环境中的决策

在本节中,我们正式介绍 POMDP 模型和相关决策概念。

12.2.1 POMDP 模型

POMDP 模型和在 1.3 节中介绍的完全可观察的 MDP 模型存在诸多相同之处,但为了表述的完整性,我们会再次提及这些相同点。时间是以步长为单位进行离散化处理的,且在每个时间步长开始时,学习器都必须执行一个动作(action)。我们只考虑离散有限的模型,这是因为目前为止 POMDP 最常用到的模型就是离散有限的,而如果采用连续模型,将很难对其进行求解。简单起见,我们假设环境表示为一个有限状态集合 $S = \{s^1, \dots, s^N\}$ 。所有可能发生的动作集合表示为 $A = \{a^1, \dots, a^K\}$ 。每一个时间步长内学习器在状态 s 下采取动作 a ,环境则根据概率转换函数 $T(s, a, s')$ 由状态 s 转变为 s' ,并且学习器会接收到一个即时回报 $R(s, a, s')$ 。

区别 POMDP 和完全可观测 MDP 之处在于,学习器在 POMDP 模型中不再直接观察状态 s' ,而是去感知一个观测(或采样) $o \in \Omega$ 。离散观测集合 $\Omega = \{o^1, \dots, o^M\}$ 代表所有可能的学习器可能接收到的传感器读数。学习器接收到的观测值取决于下一个环境状态 s' 以及在一定条件下可能也取决于学习器自身采取的动作 a 。我们可以通过这样的观测函数来刻画

这一过程: $O: S \times A \times \Omega \rightarrow [0,1]$ 。在状态 s' 执行动作 a 后观测到 o 的概率是 $O(s', a, o)$ 。为使观测函数 O 成为一个在可能的观测上的合理概率分布, 需满足, $\forall s' \in S, a \in A, o \in \Omega, O(s', a, o) \geq 0$ 同时满足 $\sum_{o \in \Omega} O(s', a, o) = 1$ 。或者, 如果观测不受学习器采取的动作的影响[⊖], 那么观测函数也可以定义为 $O: S \times \Omega \rightarrow [0,1]$ 。

在 MDP 中, 学习器的目标是通过动作来实现长期的回报最大化, 例如:

$$E \left[\sum_{t=0}^h \gamma^t R_t \right] \quad (12.1)$$

其中 $E[\cdot]$ 表示期望算子, h 表示规划域, γ 是阻尼系数, 且满足 $0 \leq \gamma \leq 1$ 。

类比于定义 1.3.1, 我们可以按如下方式定义 POMDP 模型。

定义 12.2.1 一个部分可观察马尔可夫决策过程 可以表示为这样的一个元组 $\langle S, A, \Omega, T, O, R \rangle$, 其中 S 表示一个有限状态集合, A 表示一个有限动作集合, Ω 表示一个有限观测集合, T 表示一个概率转换函数 $T: S \times A \times S \rightarrow [0,1]$, O 是一个观测函数 $O: S \times A \times \Omega \rightarrow [0,1]$, R 是一个回报函数 $R: S \times A \times S \rightarrow \mathbb{R}$ 。

图 12.1 通过对一个 POMDP 学习器与环境交互动作进行示例性的表述来说明上述这些概念。

为了说明观测函数是如何对不同类型的部分可观察性进行建模的, 我们需要考虑以下案例, 这些案例假设存在一个包含 2 个状态、2 个观测、1 个动作 (由于只有 1 个动作因此出于描述的简单起见而直接忽略这个动作) 的 POMDP 模型。在这种情形下可以对传感器的出错或噪声进行如下的建模, 例如:

$$O(s^1, o^1) = 0.8, O(s^1, o^2) = 0.2, O(s^2, o^1) = 0.2, O(s^2, o^2) = 0.8$$

是对一个装备了测量正确率为 80% 的传感器的学习器的观测情形建立的模型。当学习器观测到 o^1 或者 o^2 , 学习器本身不确定环境是处在状态 s^1 还是 s^2 。将同一观测应用于两个状态 (此时另一个观测在有效性上是冗余的), 就能对任何一个状态是否对学习器完全隐藏的概率进行建模:

$$O(s^1, o^1) = 1.0, O(s^1, o^2) = 0.0, O(s^2, o^1) = 1.0, O(s^2, o^2) = 0.0$$

390

当学习器接收到观测 o^1 时, 它无法分辨环境是否在 s^1 状态还是在 s^2 状态, 上述方法就能对隐藏状态进行充分合理的建模。

12.2.2 连续和结构化的表达

正如前文所述, 在本章中展示的算法是基于离散 POMDP 模型的。在这种前提条件下, 状态、动作以及观测空间均可以用有限集合来表示。现在我们简要地探讨一下连续和结构化的 POMDP 的表达, 这样的表达虽然不常见, 但在某些应用中有时也会用到。

有些真实世界中的 POMDP 问题采用连续模型进行建模要更恰当些 [Porta et al, 2006; Brunskill et al, 2008]。比如一个机器的姿态一般是用连续坐标 (x, y, θ) 来表示的。在标准解决方案中值迭代法也是定义在连续状态空间中的 [Porta et al, 2005]。另外, 连续观测空间 [Hoey and Poupart, 2005] 以及连续性动作 [Spaan and Vlassis, 2005b] 也有相关的研究进展。然而, 定义在连续空间中的信度、观测、动作以及回报模型存在任意种可能的不可参数化

⊖ 技术上来说, 通过将上一个动作 a 视作状态 s' 的一个特征, 我们也可以做到仅用 s' 和 o 就能确定最终观测, 即用 $O(s', o)$ 来取代 $O(s', a, o)$ 表达同样的模型。

的表达形式。为了设计出一个合适的算法，最好还是选择那些具有简单参数化形式且最终能得到闭合信度更新（closed belief update）以及贝尔曼备份的模型。例如，高斯混合或者基于粒子的表达都可用在对模型进行信度表达和高斯线性组合 [Porta et al, 2006]。作为一个备选项，基于仿真的方法通常能够处理连续状态和连续动作空间 [Thrun, 2000; Ng and Jordan, 2000; Baxter and Bartlett, 2001]。

让我们回到有限模型的话题。在很多领域中，一个更加结构化的 POMDP 表示相较于一个平坦的表示（flat representation）而言要更为有利（在平坦表示中，所有的集合都会被枚举）。动态贝叶斯网络（Dynamic Bayesian Network）通常会用于表示一个因子化（factored）的 POMDP [Boutilier and Poole, 1996; Hansen and Feng, 2000]。另外，代数决策图可以生成紧凑模型和基于规则的表示形式 [Poupart, 2005; Shani et al, 2008]。在第 8 章讲述到的关联表示也可用于表示 POMDP 模型 [Sanner and Kersting, 2010; Wang and Khordon, 2010]。此外，在某些问题中，采用多层级的结构化决策树（见第 9 章）能够提升可扩展性 [Pineau and Thrun, 2002; Theodorou and Mahadevan, 2002; Foka and Trahanias, 2007; Sridharan et al, 2010]。最后在某些情况下，当多学习器在一个部分可观察且随机的环境中同时执行一个联合任务的时候，去中心化的 POMDP 模型更为适用 [Bernstein et al, 2002; Seuken and Zilberstein, 2008; Oliehoek et al, 2008]，这一点可以参见第 15 章。

12.2.3 优化决策记忆

如 12.2.1 节中示例所述，在一个 POMDP 问题中，学习器的观测并不能唯一地决定当前的环境状态。尽管如此，由于回报本身依然是和环境状态以及状态转换相关的，一个单一的观测并不是一个马尔可夫状态信号。特别地，一个从观测到动作的直接映射是不足以作为最优动作的。为了使学习器在部分可观察环境中成功地选择它的动作，决策记忆十分必要的。

为了说明这一点，我们考虑采用双状态的有限域 POMDP 问题，如图 12.2 所示 [Singh et al, 1994]。学习器拥有两个动作，其中一个动作会按照规则将学习器传输到环境的另一个状态，而执行另一个动作将不会影响学习器所处的状态。如果学习器跳转到另一个状态，那么它将收到一个回报 $r > 0$ ，或者惩罚 $-r$ 。基础

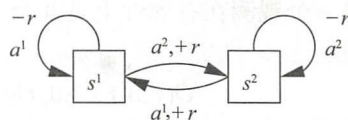


图 12.2 一个双状态 POMDP [Singh et al, 1994]，其中学习器在两个不同环境状态下接收到同样的观测

MDP 的最优策略在每一个时间步长内都能获得的回报是 $\frac{r}{1-\gamma}$ 。然而在 POMDP 问题中，学习器在两个不同的状态下收到同样的观测。这将导致仅存在两种可能的无记忆确定性静态策略（memoryless deterministic stationary policies）：要么总是执动作 a^1 要么总是执动作 a^2 。在这样的策略下，最大期望回报是 $r - \frac{\gamma r}{1-\gamma}$ ，前提条件是学习器成功地在第一个时间步长内跳转到了另一个状态。如果我们允许使用随机性（即非确定性）策略，那么最好的静态策略将得到一个值为 0 的期望减量回报（expected discounted reward），其策略是随机的以 50% 的概率选取动作 a^1 或作 a^2 。然而如果学习器可以记住它曾经执行过的动作，那么它可以执行一个策略使其有根据地在两个动作之间抉择。这样的一个基于记忆的策略在最差的情形下得到的回报是 $\frac{\gamma r}{1-\gamma} - r$ ，这已经很接近 MDP 问题的最优值了 [Singh et al, 1994]。

本示例说明了记忆对于考虑 POMDP 问题的最优策略的必要性。一个直截了当的实现记忆机制的方法是存储记录学习器执行过的动作序列以及接收到的观测序列。然而，这种形式的记忆显然是无穷无尽的，这对长期规划域而言是很不现实的。幸运的是，存在一个更好的选项，我们可以将 POMDP 转变为一个信度 - 状态 MDP 问题，即学习器通过采用一个信度向量 $b(s)$ 来总结所有的历史信息 [Stratonovich, 1960; Dynkin, 1965; Åström, 1965]。这种转变要求转换和观测函数都必须对学习器来说是已知的，而且只能应用于基于模型的 RL 方法中。

信度 b 是一个在状态集合 S 上的概率分布，它充当了整个规划任务的一个马尔可夫过程的信号。给定合适的状态空间，信度将是一个对历史信息充分的统计。这意味着学习器在这个统计下已经能表现最佳了^①。所有的信度包含在一个 $(|S|-1)$ 维度的单纯形 $\Delta(S)$ 中。另外我们可以利用 $(|S|-1)$ 个数来表示一个信度。每一个 POMDP 都假设存在一个初始的信度 b^0 ，它可以设置成一个在所有状态上的均匀分布（这表示我们不考虑初始环境状态）。每次迭代，学习器都采取一个动作 a 并且接收到观测 o ，其信度将由贝叶斯法则来更新：

392

$$b^{ao}(s') = \frac{p(o|s',a)}{p(o|b,a)} \sum_{s \in S} p(s'|s,a)b(s) \quad (12.2)$$

其中 $p(s'|s,a)$ 和 $p(o|s',a)$ 分别由模型的参数 T 和 O 定义，且

$$p(o|b,a) = \sum_{s' \in S} p(o|s',a) \sum_{s \in S} p(s'|s,a)b(s) \quad (12.3)$$

是一个正则化常量。

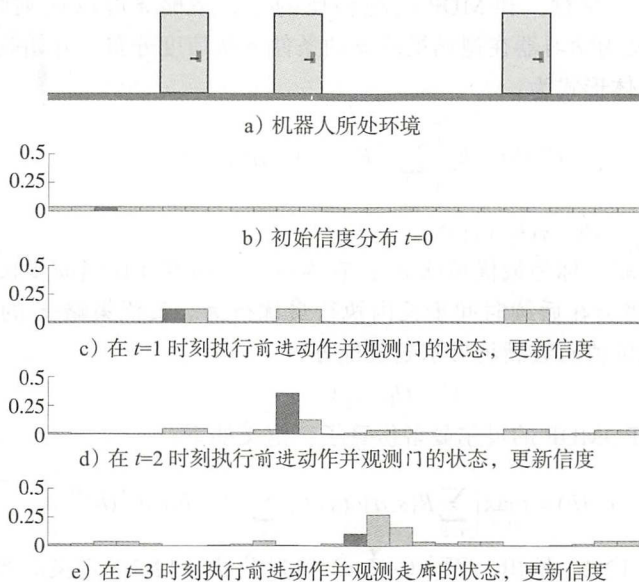


图 12.3 信度更新示例 (改编自 [Fox et al, 1999])。a) 表示一个机器人在一个拥有 3 个门的走廊上运动；b) ~ e) 表示随着时间的推移，信度分布的更新过程

图 12.3 展示了一个示例，它描述了一个机器人在有着 3 个完全相同的门的走廊上进行寻路的场景下其信度分布更新的序列。走廊被离散化为 26 个状态并且构成了一个环状，也

① 由信息学的香农理论可知在足够高的采样密度下，统计可以还原原始数据的全部信息——译者注。

就是说,当机器人走到右侧尽头的时候继续前行会回到走廊的最左边。机器人可以观测到当前是门还是走廊,但是它上面的传感器接收的数据存在噪声,即有可能会得到错误的观测结果。当一个机器人被放置在一个门的前面时,它检测到是门的概率是 0.9,换言之它的传感器出错的概率是 0.1。同样的,当机器人置于没有门的走廊上的时候,它正确检测到是走廊的概率也是 0.9。机器人可以做两个动作:前进和后退(对应于图中的向右和向左)。机器人前行或后退的时候自身位置的改变也是一个概率分布,有 20% 的概率会沿动作方向移动 3 个状态,60% 的概率会移动 4 个状态,剩下 20% 的概率会移动 5 个状态。初始的信度分布 b^0 是均匀的,如图 12.3b 所示。图 12.3c 至 (e) 展示的是机器人信度的更新。真实的机器人所处的位置可以通过暗灰色的信度分布条得知。在图 12.3c 中,我们可以看到机器人位于第一个门的前面,尽管它很确信自己位于门的前面,但是它却无法知道到底是哪一个门(因为每一扇门都是一模一样的,传感器无法区分)。然而,当机器人继续前行到另一扇门的时候,它就能依据自己动作和观测的历史纪录得到关于这个门的信息了(比如由于上一次观测到门与下一次观测到门之间需要移动的间隔数不同,可以用于区分不同的门(见图 12.3d))。但到了图 12.3e 的时候,信度分布又开始变得模糊了,这是因为状态迁移模型包含噪声,而且观测到走廊这个事件在本例中信息基本是无效的(观测存在噪声且走廊占比很大)。

12.2.4 策略和价值函数

正如完全可观测 MDP 问题所言,学习器的目标是选择动作使得任务尽可能的完美地完成。也就是要让学习器学习一个最优策略。在 POMDP 问题中,一个最优策略 $\pi^*(b)$ 将信度映射到动作上。注意,和 MDP 问题相反的是,策略 π 可以被刻画成一个价值函数 $V^\pi: \Delta(S) \rightarrow \mathbb{R}$, 定义为学习器在遵循策略 π 的条件下从信度分布 b 开始收集到的期望未来减量回报 $V^\pi(b)$, 其具体形式为:

$$V^\pi(b) = E_\pi \left[\sum_{t=0}^h \gamma^t R(b_t, \pi(b_t)) \mid b_0 = b \right] \quad (12.4)$$

其中 $R(b_t, \pi(b_t)) = \sum_{s \in S} R(s, \pi(b_t)) b_t(s)$ 。

最大化 V^π 的策略 π 称为最优策略 π^* ; 它为每一个信度 b 在当前步长内执行一个最优行为,并假设学习器将会在后续时间步长内执行最优行为。最优策略 π^* 的值是通过最优价值函数 V^* 定义的。该价值函数满足贝尔曼最优方程

$$V^* = H_{\text{POMDP}} V^* \quad (12.5)$$

其中 H_{POMDP} 是 POMDP 的贝尔曼备份算子,定义如下:

$$V^*(b) = \max_{a \in A} \left[\sum_{s \in S} R(s, a) b(s) + \gamma \sum_{o \in O} p(o \mid b, a) V^*(b^{ao}) \right] \quad (12.6)$$

其中 b^{ao} 通过公式 (12.2) 给出,而 $p(o \mid b, a)$ 通过公式 (12.3) 定义。当公式 (12.6) 对每个 $b \in \Delta(S)$ 都是满足的时候,我们就能确信该解决方案是最优的。

在一个连续信度空间中计算价值函数可能看起来很棘手,但幸运的是,价值函数有一个特殊的可以利用的结构 [Sondik, 1971]。价值函数可以通过有限数量的向量进行参数化,且属于下凸函数。价值函数的下凸函数性质暗示其信度靠近信度单纯形的两端的值将会较高。通常来说,学习器对于自身的真实状态的不确定越少,那么它对未来的预测效果会更好,进而做出更好的决策。一个位于某个信度单纯形 $\Delta(S)$ 两端点上的信度,换句话说,对于某个

特定的 s 有 $b(s)=1$ ，它代表当前的学习器所处状态是完全确定的。通过这种方式，价值函数 V 的下凸性质就能得到初步的解释了。一个双状态的 POMDP 问题的下凸价值函数的示例如图 12.4a 所示。由于信度空间是单纯形，因此我们可以将一个双状态的 POMDP 问题的任意信度都表示在一条线上，因为总是满足 $b(s^2)=1-b(s^1)$ 。信度单纯形的两个端点标记为 $(1,0)$ 和 $(0,1)$ 。显而易见，两端点的信度要高于信度空间的中间部分，例如 $(0.5,0.5)$ 。

另一个表示 POMDP 问题中的策略的途径是考虑策略树 [Kaelbling et al, 1998]。图 12.4b 展示了一个策略树的部分。学习器一开始从树的根节点出发。每一个节点对应于一个学习器处于该节点时所执行的动作。下一步，学习器接收到一个观测 o ，它决定学习器将会转换到下一级的哪个节点。树的深度取决于规划域 h ，也就是说，如果我们希望学习器考虑执行 h 步，那么对应的策略树的深度也是 h 。

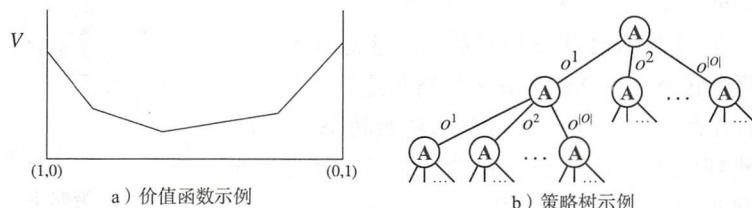


图 12.4 a) 展示了一个双状态的 POMDP 的价值函数。其中 y 轴表示信度大小， x 轴表示信度空间 $\Delta(S)$ ，范围从 $(1,0)$ 到 $(0,1)$ ；b) 展示了一个策略树的示例，每当学习器执行一个动作，学习器就会根据接收到的观测 $o \in \{o^1, o^2, \dots, o^{|O|}\}$ 从当前所处节点跳转到下一级的某个节点

12.3 基于模型的技术

如果学习器所处环境是可以建模的，那么可以用这个环境模型为学习器计算一个策略。在本节中，我们将讨论几种计算 POMDP 策略的方法，这些方法包含最优、近似最优以及启发式方法。尽管环境对学习器来说是完全已知的，但目前只能对小规模的 POMDP 问题的最优解进行计算，另外在考虑采用哪种方法时，真正让人感兴趣的是如何在最优性和效率之间进行折衷处理。本节中展示的所有方法均采用了一个信度状态表示形式（见 12.2.3 节），因为它提供了一个对完整学习器动作进程历史的紧凑的表达形式。

395

12.3.1 基于 MDP 的启发式解决方案

首先，我们讨论一些启发式控制策略，这些提出的控制策略依赖于基础 MDP 的解 $\pi_{\text{MDP}}^*(s)$ 或者 $Q_{\text{MDP}}^*(s,a)$ [Cassandra et al, 1996]。求解 MDP 的计算复杂度比求解 POMDP 要低得多（前者是 P 完全问题，后者是 PSPACE 完全问题）[Papadimitriou and Tsitsiklis, 1987]，但通过追踪信度状态，某些形式的不完美的状态感知也是可以得到修正的。[Cassandra, 1998] 提供了一个广泛的基于 MDP 的各启发式技术之间的实验性比较。

大概最直接的启发式技术是为一个给定的时间步长上的信度考虑其最可能的状态 (MLS)，并且 MDP 策略为该状态指定的要执行的动作是

$$\pi_{\text{MLS}}(b) = \pi_{\text{MDP}}^*(\arg \max_s b(s)) \quad (12.7)$$

MLS 启发式技术完全忽略了当前信度的不确定性，这显然是次优的。

一个更为复杂的逼近技术是 Q_{MDP} [Littman et al, 1995], 它将 POMDP 问题当作完全可观测的方式进行处理。 Q_{MDP} 对 MDP 问题进行求解并定义了一个控制策略

$$\pi_{Q_{\text{MDP}}}(b) = \arg \max_a \sum_s b(s) Q_{\text{MDP}}^*(s, a) \quad (12.8)$$

Q_{MDP} 在某些领域非常有效, 但是由它计算出来的策略所执行的动作没有充分地利用信息, 正如 Q_{MDP} 的解决方案假设的那样: 任何和状态相关的不确定性都在执行了某个动作后就消失了。鉴于此, Q_{MDP} 策略在应对需要进行重复信息收集的场景中会失效。

例如, 考虑如图 12.5 所示的玩具领域 (toy domain), 它展示了基于 MDP 的启发式方法是如何在该领域中失效的 [Parr and Russell, 1995]。学习器从状态 I 出发, 然后无论采取什么动作都以等概率转换到两个状态中的任何一个。在两个状态下, 学习器都会接收到观测 A 。这意味着学习器无法分辨这两个状态。POMDP 最优策略是连续执行一个动作 a 两次, 而在此之后, 学习器将返回初始状态。然而, 由于学习器观测到的要么是 C 要么是 D , 所以它知道当前被标记为 A 的到底是两个状态中的哪一个。而这一知识对于学习器选取最优行为 (b 或 c) 以实现正回报 (标记为 $+1$) 的状态转换而言是很重要的。动作 a 并不会改变系统状态, 但却会改变学习器的信度状态 (两个时间步长后), 这一事实对于基于 MDP 的方法来说是相当难以规划的。这个事实促成了对学习器动作执行效果收集的明确信息的推理, 而这一点对于基于 MDP 的解决方案来说是难以做到的。

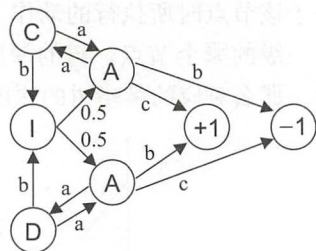


图 12.5 一个在基于 MDP 的控制策略下会失效的简单场景

我们还可以对 MDP 的设置进行扩展以实现某种形式的感知不确定性进行建模, 且不必考虑所有的 POMDP 信度。例如, 在机器人技术中, 在定位包含不确定性的问题下的机器人导航是可以通过信度分布的均值和熵进行建模的 [Cassandra et al, 1996; Roy and Thrun, 2000]。尽管从计算复杂度低这一点来看, 基于 MDP 的方法确实很吸引人, 但是这种方法很可能在当信度分布并非为单一模式而是更为复杂的形状模式的时候而失效。

12.3.2 POMDP 的值迭代

为了克服基于 MDP 的启发式方法所带来的局限性, 我们现在考虑通过值迭代的方法来计算最优的 POMDP 策略。采用信度状态后, 我们可以将原来的离散状态 POMDP 问题转化为连续状态的 MDP 问题。前面已经论述了通过价值函数可以表示一个 MDP 问题的规划。价值函数为每个状态估计学习器根据某个规划执动作所能收集到的总体减量累计回报值。在一个 POMDP 问题中, 最优价值函数, (即与最优规划相对应的价值函数) 展示了一个特殊的结构 (它在自身的每一分段上是线性的, 而整体又是下凸的函数), 可利用这个结构来帮助计算问题的解。举例而言, 值迭代方法是一种为了对 POMDP 问题进行求解而建立一系列的价值函数的估计, 使得这些估计最终收敛于当前任务的最优价值函数 [Sondik, 1971]。一个有限域 POMDP 问题的价值函数是由有限个处于信度空间中的超平面或向量实现其参数化的。这些向量将信度空间划分成有限个数的区域。每一个向量都在某个区域中最大化价值函数, 并对应于一个在该信度空间区域内最优的动作。

正如我们即将要解释的, 计算下一个价值函数的估计 (即往未来的方向考虑更深一步) 需要考虑一个学习器所有可能执行的动作以及所有可能接收到的观测。不幸的是, 这将导致

当规划域增大时向量个数的指数式增长。很多计算过的向量都会变得无用，因为它们所对应的最大化信度空间区域将是空的，但验证并紧接着净化所有这些向量（指剔除那些无用的向量）的工作是一项代价昂贵的操作。

确切的值迭代算法 [Sondik, 1971; Cheng, 1988; Cassandra et al, 1994] 在每一个值迭代步骤中搜索完整的信度单纯形以寻找一个能为下一规划域的价值函数产生必要的向量集合的最小信度点集。这通常需要线性规划（linear programming），因此当维度很高的时候，计算代价会很高。其他额外的值迭代算法侧重于生成所有可能的下一规划域的向量，紧接着或者在同时采用一种智能的方法来净化占主导地位的向量 [Monahan, 1982; Zhang and Liu, 1996; Littman, 1996; Cassandra et al, 1997; Feng and Zilberstein, 2004; Lin et al, 2004; Varakantham et al, 2005]。尽管如此，净化向量依然需要采用线性规划。

最优策略 π^* 的值是由最优价值函数 V^* 定义的，而最优价值函数是通过迭代一定步骤后计算得到的，在每一个步骤里都考虑未来下一步的情形。在每次迭代中，我们应用确切的动态规划算子 H_{POMDP} （公式（12.6））。如果学习器只能再执行一个动作时，我们仅仅为信度分布 b 考虑学习器的立即回报，并且此时可以忽略任何未来价值函数的值。公式（12.6）此时也可以归约为：

$$V_0^*(b) = \max_a \left[\sum_s R(s, a) b(s) \right] \quad (12.9)$$

我们将立即回报函数 $R(s, a)$ 当作一个包含 $|A|$ 个向量的集合 $\alpha_0^a = (\alpha_0^a(1), \dots, \alpha_0^a(|S|))$ ，每一个分别对应于动作 a ： $\alpha_0^a(s) = R(s, a)$ 。现在我们重写公式（12.9）为下列形式，其中把 b 当作一个 $|S|$ 维度的向量：

$$V_0^*(b) = \max_a \sum_s \alpha_0^a(s) b(s) \quad (12.10)$$

$$= \max_{\{\alpha_0^a\}_a} b \cdot \alpha_0^a \quad (12.11)$$

其中 (\cdot) 代表内积操作。

在一般的情形下，对于 $h > 0$ ，我们在第 n 次迭代时对一个价值函数 V_n 进行参数化，其参数是一个有限向量或超平面集合 $\{\alpha_n^k\}, k=1, \dots, |V_n|$ 。在第 n 步给定一个向量集合 $\{\alpha_n^k\}_{k=1}^{|V_n|}$ ，其对应于信度 b 的值由下式给出

$$V_n(b) = \max_{\{\alpha_n^k\}_k} b \cdot \alpha_n^k \quad (12.12)$$

另外，动作 $a(\alpha_n^k) \in A$ 和每一个向量关联，且是当前基于 α_n^k 是最大化向量的信度而执行的最优行为。每一个向量定义了一个信度空间区域，且该向量是 V_n 中最大的元素。这些区域构成了信度空间的一个分区，价值函数的分段线性的性质使得它们是可以归约的，如图 12.6 所示。

价值函数在 b 的梯度由以下向量给出

$$\alpha_n^b = \arg \max_{\{\alpha_n^k\}_k} b \cdot \alpha_n^k \quad (12.13)$$

且在 b 处的策略由下式给出

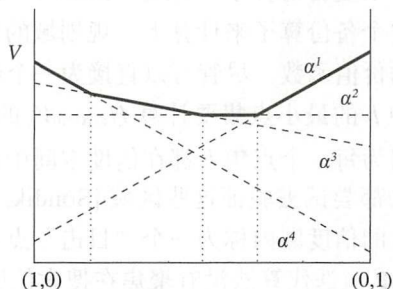


图 12.6 一个具体的 POMDP 价值函数示例，请对照图 12.4a。图中价值函数是指实线部分，它由 4 个 α 向量构成，每一个向量对应图中的一条虚线。信度空间的划分如垂直的虚线所示

$$\pi(b) = a(\alpha_n^b) \quad (12.14)$$

POMDP 问题的值迭代算法背后的主要思想是, 对于一个给定的价值函数 V_n 和一个特别的信度点集 b , 我们可以很容易地计算出 $H_{\text{POMDP}} V_n$ 的向量 α_{n+1}^b , 且有

$$\alpha_{n+1}^b = \arg \max_{\{\alpha_{n+1}^k\}_k} b \cdot \alpha_{n+1}^k \quad (12.15)$$

其中 $\{\alpha_{n+1}^k\}_{k=1}^{|H_{\text{POMDP}} V_n|}$ 是 $H_{\text{POMDP}} V_n$ 的未知向量集合。公式 (12.15) 记为 $\alpha_{n+1}^k = \text{backup}(b)$ 。为此, 我们定义向量 g_{ao} 为

$$g_{ao}^k(s) = \sum_{s'} p(o|s', a) p(s'|s, a) \alpha_n^k(s') \quad (12.16)$$

它表示对特定的动作 a 和观测 o 反投射 α_n^k 所得到的向量。由公式 (12.6) 可以推导出:

$$V_{n+1}(b) = \max_a \left[b \cdot \alpha_0^a + \gamma b \cdot \sum_o \arg \max_{\{g_{ao}^k\}_k} b \cdot g_{ao}^k \right] \quad (12.17)$$

$$= \max_{\{g_a^b\}_a} b \cdot g_a^b \quad (12.18)$$

$$\text{with } g_a^b = \alpha_0^a + \gamma \sum_o \arg \max_{\{g_{ao}^k\}_k} b \cdot g_{ao}^k \quad (12.19)$$

也可以写作如下形式:

$$V_{n+1}(b) = b \cdot \arg \max_{\{g_a^b\}_a} b \cdot g_a^b \quad (12.20)$$

从公式 (12.20) 可以推导得到向量 $\text{backup}(b)$, 因为该向量和信度点集 b 的内积可以得到 $V_{n+1}(b)$:

$$\text{backup}(b) = \arg \max_{\{g_a^b\}_{a \in A}} b \cdot g_a^b \quad (12.21)$$

其中 g_a^b 的定义参考公式 (12.19)。注意到在一般情况下, 不仅仅计算过的 a 向量保存了下来, 而且动作 a 在公式 (12.21) 中是一个最大化子, 这是由于最优行为和向量 $\text{backup}(b)$ 相关。

12.3.3 确切的值迭代

贝尔曼备份算子 (公式 (12.21)) 计算每一个信度下的下一规划域的向量, 现在我们将采用这个备份算子来计算下一规划域的完整价值函数, 即, 计算对信度空间中的所有信度都最优的价值函数。尽管可以直接为一个给定的 b 计算其对应的向量 $\text{backup}(b)$, 但搜索信度空间中 b 的最小点集要计算 $H_{\text{POMDP}} V_n$ 的所有向量 $\cup_b \text{backup}(b)$, 这个步骤的计算代价非常高。因为每一个点集 b 都在信度空间中有一个区域, 在这个区域中 α_n^b 能达到最大值, 一系列的算法都尝试去验证这些区域 [Sondik, 1971; Cheng, 1988; Kaelbling et al, 1998]。对应的 b 在各自的信度区内称为一个“目击”点, 因为它证明了它所在的区域的存在性。别的确切的 POMDP 值迭代算法没有聚焦在搜索信度空间上。相反, 这些算法考虑枚举 $H_{\text{POMDP}} V_n$ 对应的所有可能的向量, 然后净化无用向量 [Monahan, 1982; Zhang and Liu, 1996; Littman, 1996; Cassandra et al, 1997; Feng and Zilberstein, 2004; Lin et al, 2004; Varakantham et al, 2005]。我们将关注这些枚举算法, 因为这些算法最近得到了更大程度上的发展且也得到更为广泛的应用。

12.3.3.1 Monahan 枚举算法

首先, 我们考虑计算 $H_{\text{POMDP}}V_n$ 的最直接的方法, 该方法由 [Monahan, 1982] 提出。它利用价值函数的已知结构来计算构建 $H_{\text{POMDP}}V_n$ 的所有可能方式。注意到在每一个可能的 $H_{\text{POMDP}}V_n$ 中, 都会生成有限数量的向量, 比如我们前面假设过的动作集合 A 和观测集合 O 。现在由于我们是独立地在某一信度 b 上进行操作, 所以原来的公式 (12.19) 以及公式 (12.21) 都没办法再使用了。我们不再寻求在某一特定信度点集 b 下对所有观测 $o \in O$ 在 g_{ao}^k 向量上的最大化, 取而代之, 我们必须考虑所有为观测 o 选择对应的 g_{ao}^k 向量的方式:

$$H_{\text{POMDP}}V_n = \bigcup_a G_a, \text{ with } G_a = \bigoplus_o G_a^o \text{ 且 } G_a^o = \left\{ \frac{1}{|O|} \alpha_0^a + \gamma g_{ao}^k \right\}_k \quad (12.22)$$

其中操作符 \oplus 表示交叉和算子^①。

不幸的是, 在每一阶段, 生成的向量集合包含有限但指数级个数的向量, 该集合记作: $|A||V_n|^{|O|}$ 。在这些生成的向量中有很多所对应的信度区域将会是空的, 因此这些向量都属于无用向量, 因为它们根本不影响学习器的策略。技术上来讲, 它们不属于价值函数的一部分, 保存这些无用向量不会对后续价值函数产生任何影响, 但却会带来计算负担。因此, 所有基于枚举的值迭代算法都采用了某种形式的净化方案。特别是 [Monahan, 1982] 提出净化 $H_{\text{POMDP}}V_n$ 的方案如下:

$$V_{n+1} = \text{prune}(H_{\text{POMDP}}V_n) \quad (12.23) \quad \boxed{400}$$

其中 $H_{\text{POMDP}}V_n$ 在公式 (12.22) 中已经定义。prune 算子是基于线性规划实现的 [White, 1991]。

12.3.3.2 增量净化

[Monahan, 1982] 提出的算法首先生成 $H_{\text{POMDP}}V_n$ 所有的 $|A||V_n|^{|O|}$ 向量, 然后对所有占主导的向量进行净化。增量净化方法 [Zhang and Liu, 1996; Cassandra et al, 1997; Feng and Zilberstein, 2004; Lin et al, 2004; Varakantham et al, 2005] 通过利用如下的事实来节约计算时间:

$$\text{prune}(G \oplus G' \oplus G'') = \text{prune}(\text{prune}(G \oplus G') \oplus G'') \quad (12.24)$$

在这种情况下, 线性规划中用于净化向量的那些约束的数量将会增长地相对较慢 [Cassandra et al, 1997], 因此能得到更好的性能。基本增量净化算法利用公式 (12.24) 来计算 V_{n+1} , 其过程如下:

$$V_{n+1} = \text{prune}\left(\bigcup_a G_a\right) \quad (12.25)$$

$$G_a = \text{prune}\left(\bigoplus_o G_a^o\right) \quad (12.26)$$

$$= \text{prune}\left(G_a^1 \oplus G_a^2 \oplus G_a^3 \oplus \dots \oplus G_a^{|O|}\right) \quad (12.27)$$

$$= \text{prune}\left(\dots \text{prune}\left(\text{prune}(G_a^1 \oplus G_a^2) \oplus G_a^3\right) \dots \oplus G_a^{|O|}\right) \quad (12.28)$$

然而, 一般而言计算 POMDP 的确切解是棘手的问题 [Papadimitriou and Tsitsiklis, 1987; Madani et al, 2003], 因此需要寻求近似 (逼近) 解的方法 [Lovejoy, 1991; Hauskrecht, 2000]。接下来我们将介绍一系列目前较为流行的基于值迭代的逼近算法。

① 集合的交叉和定义为: $\bigoplus_k R_k = R_1 \oplus R_2 \oplus R_3 \oplus \dots \oplus R_K$, 其中 $P \oplus Q = \{p+q | p \in P, q \in Q\}$ 。

12.3.4 基于点的值迭代方法

由于求解 POMDP 问题的最优解拥有很高的计算复杂度, 许多 POMDP 近似最优解的解决方案应运而生。其中一个很给力的想法是仅针对处于信度单纯形上的可达 (即在学习器与环境交互中能遇到) 的那部分进行计算问题的解。这一想法激发了采用近似解技术辅助求解的方法, 该方法创新之处在于使用了一个基于信度点的采样集合, 且采样区仅选择学习器动作规划所执行的那部分信度空间区域 [Hauskrecht, 2000; Poon, 2001, Roy and Gordon, 2003; Pineau et al, 2003; Smith and Simmons, 2004; Spaan and Vlassis, 2005a; Shani et al, 2007; Kurniawati et al, 2008], 这一方法的其中一种可能的实现已经由 [Lovejoy, 1991] 提出。其思路是不再在完整的学习器信度空间上执行规划 (对于大规模的状态空间, 在对应的信度空间上执行规划将会变得很棘手), 取而代之的是我们只需要在一个大小受限的原型信度集合 B 上执行规划, 该原型信度集合是通过学习器和环境之间进行交互从而采样得到的。

正如前文所述, 导致 POMDP 问题求解变得棘手的一个主要原因是这些求解方法都试图对信度空间 $\Delta(S)$ 中所有可能的信度计算其对应的最优行为。例如, 如果我们采用 Monahan 的算法 (公式 (12.22)), 我们最终会生成一系列的价值函数, 这些价值函数的大小将随着规划域的增大而呈现指数级增大。一个自然的破解当前局面的方法是考虑仅利用一个有限大小的信度点集取代完整信度空间来求 POMDP 问题的近似最优解。在备份阶段 (即贝尔曼备份算子应用阶段), 我们将以固定的次数反复应用公式 (12.21), 最终得到数量很少的向量 (数量上界是信度集合的大小)。利用近似方法的动机在于近似方法能够对更加庞大规模的问题进行计算并且能得到成功的策略, 其计算的高效性弥补了它不一定得出最优解的缺点。

基于点的方法的总体假设是, 通过更新在信度 $b \in B$ 上的值以及其梯度 (也就是 α 向量), 最终计算得到的策略将具有很好的泛化能力, 而且对于即便是处于信度点集 B 之外的情形依然有效。无论上述假设对于 POMDP 的结构和信度 B 的内容来说是否现实, 但凭直觉, 在诸多的问题中, 这个“可达”信度集合 (这里的“可达”指的是从初始信度开始经由任意一个动作策略, 信度分布不断更新时可能达到指定信度状态) 在信度单纯形中形成一个低维流形, 因此一个相对小规模的信度点集就能足够密集地覆盖“可达”信度集合。

基本基于点的 POMDP 更新操作如下。该算法利用一个近似备份算子 \tilde{H}_{PBVI} 来取代原来的 H_{POMDP} , 在每一个值备份阶段, 需要计算下列集合

$$\tilde{H}_{\text{PBVI}} V_n = \bigcup_{b \in B} \text{backup}(b) \quad (12.29)$$

其中需要使用到一个固定的信度点集 B 。另一种方案是随机化备份操作算子 $\tilde{H}_{\text{PERSEUS}}$, 它由 PERSEUS 算法给出 [Spaan and Vlassis, 2005a]。该算子增加 (或至少不会降低) 信度点集 B 中的所有信度的值。其关键思想是在每一个值备份阶段, 所有信度点的值都可以仅通过备份一个随机选取的信度点子集 \tilde{B} 得到相应的提高:

$$\tilde{H}_{\text{PERSEUS}} V_n = \bigcup_{b \in \tilde{B}} \text{backup}(b), \quad (12.30)$$

$$\text{确保 } V_n(b') \leq V_{n+1}(b'), \forall b' \in B \quad (12.31)$$

在每一个备份阶段, 信度点子集 \tilde{B} 通过对 B 进行采样得到, 采样的终止条件是当在信度点集 B 上的价值函数 V_{n+1} 的值的上界都不低于 V_n , 也就是说直到公式 (12.31) 得到满足。 $\tilde{H}_{\text{PERSEUS}}$ 算子计算得到的价值函数有相对较少的向量, 这允许我们使用更大的信度空间 B , 进而提高 (至少不会降低) 近似最优解的精度 [Pineau et al, 2003]。

计算出的近似解的控制质量的关键在于 B 的构成。目前已经提出一些构建 B 的方案。例如, 可以使用由诸如 Freudenthal 三角测量 [Lovejoy, 1991] 之类的方法计算得到信度单纯形上的常规网格。其他选择包括采取所有的信度单纯形的极值点或使用随机网格 [Hauskrecht, 2000; Poon, 2001]。一个替代方案是包含通过模拟 POMDP 可以遇到的所有信度点: 我们可以通过在每个时间步长上对学习器的随机动作和观测进行采样, 来产生信度空间中的信度变化轨迹 [Lovejoy, 1991; Hauskrecht, 2000; Poon, 2001; Pineau et al, 2003; Spaan and Vlassis, 2005a]。该采样方案将 B 的内容重点放在 POMDP 模型执行过程中实际经历过的信度上。

402

也提出了更复杂的信度采样方案。例如, 可以使用 MDP 解决方案来指导信度采样过程 [Shani et al, 2007], 但是在需要一系列信息收集动作的问题领域中, 启发式方法将遇到与使用 Q_{MDP} 时类似的问题 (12.3.1 节)。此外, 信度集 B 不必是静态的, 并且可以在运行基于点的求解器时进行更新。基于最优价值函数的上限和下限, HSVI 从初始信念开始, 从搜索树中启发性地选择信念点 [Smith and Simmons, 2004, 2005]。算法 SARSOP 通过连续逼近最优可达信度空间 (即通过遵循最优策略可以达到的信度空间), 进一步推动了这一想法 [Kurniawati et al, 2008]。

总的来说, 基于点的计算解方法一般依据的是价值函数的分段线性和下凸性质。当给定某一信度, 学习器可以很简单地利用公式 (12.14) 搜索到学习器采取的到底是哪个动作。

12.3.5 其他近似求解方法

除了基于点的方法外, 其他类型的近似解结构也得到了进一步的探索。

12.3.5.1 基于网格的近似求解方法

避免确切 POMDP 值迭代的难处理性的一种方法是使用固定网格 [Drake, 1962; Lovejoy, 1991; Bonet, 2002] 或可变网格 [Brafman, 1997; Zhou and Hansen, 2001]。对每个网格点执行值备份, 但仅保留每个网格点的值, 并忽略渐变。非网格点的值由插值规则定义。基于网格的方法主要取决于网格点的选择以及插价值函数所采用的形状。一般来说, 常规网格在高维度问题中不能很好地进行扩展, 并且非常规网格的插值方法往往计算代价很高。

12.3.5.2 策略搜索

计算 (近似) 价值函数的另一种方法是策略搜索: 这些方法在限制类控制器中搜索一个良好的策略 [Platzman, 1981]。例如, 策略迭代 [Hansen, 1998b] 和有界策略迭代 (BPI) [Poupart and Boutilier, 2004] 通过执行策略迭代步骤搜索 (有界大小) 随机有限状态控制器的空间。搜索策略空间的其他选项包括梯度上升 [Meuleau et al, 1999a; Kearns et al, 2000; Ng and Jordan, 2000; Baxter and Bartlett, 2001; Aberdeen and Baxter, 2002] 和启发式方法, 如随机局部搜索 [Braziunas and Boutilier, 2004]。特别地, PEGASUS 方法 [Ng and Jordan, 2000] 通过使用固定的随机种子模拟来自 POMDP 的 (有界) 数量的轨迹来估计策略的值, 然后在策略空间中执行以使该值最大化。策略搜索方法在几种情况下已经取得了成功, 但在策略领域中的搜索常常是困难的, 且容易出现局部最优等问题 [Baxter et al, 2001]。

403

12.3.5.3 启发式搜索

求解 POMDP 的另一种方法是基于启发式搜索 [Satia and Lave, 1973; Hansen, 1998a; Smith and Simmons, 2004]。将初始置信度 b_0 定义为根节点, 这些方法构建一个分支 (a ,

o) 对的树, 每个树递归地引入新的信度节点。分支限界技术用于维护搜索树中边缘节点上的预期返回值的上限和下限。[Hansen, 1998a] 提出了一种将策略当作有限状态控制器进行处理的策略迭代方法, 其使用信念树将搜索集中在控制器最有可能改进的信度空间的区域。然而, 它对大问题的适用性受限于使用完整的动态规划更新。如前所述, HSVI[Smith and Simmons, 2004, 2005] 是一种近似的值迭代技术, 在信度空间中进行启发式搜索, 以便更新边界, 类似于 [Satia and Lave, 1973] 的研究。

12.4 无先验模型的决策

当学习器没有可用的环境模型时, 上一节中介绍的先验的、基于模型的方法无法直接应用。即使相对简单的技术(如 QMDP (见 12.3.1 节))也需要了解完整的 POMDP 模型: 使用转换和回报模型计算基础 MDP 的解决方案, 另外信度更新(公式(12.2))还需要对观测进行建模。

一般来说, 存在着两种解决这种决策问题的方法, 即直接和间接的强化学习方法。直接方法应用真正的无模型技术, 它们不会尝试重建未知的 POMDP 模型, 而是比如, 将观测历史直接映射到动作。另一方面, 可以尝试通过与环境进行交互来重建 POMDP 模型, 然后原则上可以使用 12.3 节中提出的技术来解决。这种间接方法长期以来一直在 POMDP 问题上不受欢迎, 因为: 1) 对 POMDP 模型的重建或其近似模型的重建是非常困难的一件事; 2) 即使采用重建后的 POMDP 模型, 基于模型的方法仍然需要花费大量时间来计算才能得到一个好的策略。然而, 基于模型的方法(例如基于信度点的一系列算法(见 12.3.4 节))的进步已经使这些类型的方法变得更具吸引力。

12.4.1 无记忆技术

首先, 我们考虑学习无记忆策略的方法, 这种无记忆策略是指直接将观测结果映射到学习器的动作上, 且不需要依赖任何系统内部状态(即不参考也不维护任何历史信息)。无记忆策略可以是确定性映射 $\pi: \Omega \rightarrow A$, 或者概率性映射 $\pi: \Omega \rightarrow \Delta(A)$ 。在 12.2.3 节中已经说明, 概率性策略能得到更高的收益, 其代价是需要维护一个不断增长以至于无法逐一枚举的搜索空间 [Singh et al, 1994]。事实上, 寻找一个最优的确定性无记忆策略是一个 NP 难问题 [Littman, 1994], 而寻找一个最优的概率性无记忆策略的算法复杂度依然难以定论。

[Loch and Singh, 1998] 经验证明, 在 SARSA (λ) 的情况下, 使用资格迹可以提高无记忆方法处理部分可观察性的能力。在几个领域中已使用 SARSA (λ) 学习最优确定性无记忆策略(可以列举所有这些策略, 其中有 $|A|^{|Q|}$ 个策略)。[Bagnell et al, 2004] 也考虑了无记忆确定性的情况, 但使用的是非静态策略而不是静态策略。其研究表明, 在不存在良好的静态策略的某些迷宫领域, 可以发现成功的非静态策略。关于学习随机无记忆策略, [Jaakkola et al, 1995] 提出了一种算法, 由 [Williams and Singh, 1999] 根据经验进行了测试, 表明该算法可以成功地学习随机的无记忆策略。层级 Q 学习 (Hierarchical Q-Learning) [Wiering and Schmidhuber, 1997] 提供了一个有趣的转折, 其目的是在 POMDP 中学习一个子目标序列, 其中每个子目标可以使用无记忆策略成功实现。

12.4.2 学习内部记忆

鉴于在没有马尔可夫状态信号(如 POMDP)的系统中无记忆策略的局限性, 研究的方

向自然转向另一方面：在每个学习器中引入了某种形式的记忆，即所谓的内部状态。存储过程的完整历史信息，即学习器采取的动作组成的向量和接收到的观测结果，这种方案由于以下几个原因而不能用于实际问题中。首先，如在没有模型的情况下，学习器无法计算信度状态，因此存储的完整历史信息将会无限制地增长。其次，这样的历史信息表示形式不容易泛化，例如，不清楚历史 $\langle a^1, o^1, a^1, o^1 \rangle$ 之后获得的经验如何更新历史 $\langle a^2, o^1, a^1, o^1 \rangle$ 的值。为了解决这些问题，研究人员提出了许多不同的内部状态表示，我们将会对它们进行一个简单的介绍。

首先，前面提到的无记忆方法可以看作保持一个单一观察的历史窗口。相反，这些算法也可以应用于包含最后 k 个观察的历史窗口 [Littman, 1994; Loch and Singh, 1998]，其中 k 通常是先验定义的参数。在某些领域（例如，策略空间相对较缓地增长（通过取一个低值 k ））可以在学习时间和任务执行性能中获得显著改进。有限历史窗口也用作神经网络的表示 [Lin and Mitchell, 1992]。

然而，有限的历史窗口不能捕获任意的长期依赖性，如图 12.7a 所示，该 T 迷宫是 [Bakek, 2002] 提供的示例。在这个问题中，学习器从 S 开始，需要导航到 G，然而，G 的位置最初是未知的，并且可能位于走廊末端的左侧或右侧。然而，在开始状态下，学习器可以对取决于特定目标位置的道路标志 X 进行观察。走廊的长度是可以变化的（在图 12.7a 中走廊长度是 10），这意味着学习器需要花费许多个时间步长才能学会记住道路标志。显然，有限历史窗口不能很好地表示这种依赖性。

为了缓解固定历史窗口的问题，[McCallum, 1993, 1995, 1996] 在其众多的研究贡献中提出了可变历史窗口的几种算法。这些技术允许历史窗口在状态空间的不同部分具有不同的深度。例如，有效化后缀记忆（Utile Suffix Memory, USM）算法通过生成一个后缀树 [McCallum, 1995] 来学习短期记忆表示，图 12.7b 展示的是其中一个例子。USM 算法对每一个实例都根据其有意义的历史信息的多少来组合这些强化学习经验。通过这种方式，在状态空间的不同部分，可以维持不同的历史长度，这一点与有限历史窗口方法是不同的。图 12.7b 中的实线描绘的是后缀树，其叶子节点是由具有匹配历史记录直到相应深度的叶子节点构成的。虚线节点是所谓的边缘节点，即算法可以考虑添加到树中的那些附加节点。当统计测试表明边缘节点的分支中的实例来自预期未来减量回报的不同分布时，使该树生长以包括该边缘分支。否则，如果添加分支将有助于预测未来的回报，则在状态空间的相应部分扩展记忆是值得的。更多关于这些想法的研究重点是在有噪声观测的情况下更好地学习动作 [Shani and Brafman, 2005; Wierstra and Wiering, 2004]。基于这些研究思路，基于长短时记忆神经网络（LSTM）架构的循环神经网络也已成功地用作内部状态表示 [Hochreiter and Schmidhuber, 1997; Bakker, 2002]。

另外存在一些其他的内部记忆表示形式。[Meuleau et al, 1999b] 扩展了 VAPS 算法 [Baird and Moore, 1999]，以学习有限状态自动机（FSA）形式的策略。FSA 表示有限策略图，其中节点代表学习器的动作，弧（边）表示观测值。如在 VAPS 中，随机梯度上升用于收敛到局部最优控制器。找到给定大小的最优策略图的问题也有相关的研究进展 [Meuleau et al, 1999a]。然而，能够合理地表示一个无限策略图才能称为最优 POMDP 策略。

最后，已提出预测状态表示（PSR）作为用于建模随机和部分可观察环境的 POMDP 的替代方案 [Littman et al, 2002; Singh et al, 2004]，见第 13 章。PSR 分配隐藏的 POMDP 状态，并且仅考虑动作和观测的序列。在 PSR 中，系统的状态在可能的未来事件序列或“核

心测试”（core test）中表示为动作和观测的交替进行。给定当前历史信息，PSR 的状态定义为可以实际实现的每个核心测试的概率向量。PSR 的优点在无模型学习设置中最为明显，因为该模型仅考虑可观测事件而不是隐藏状态。

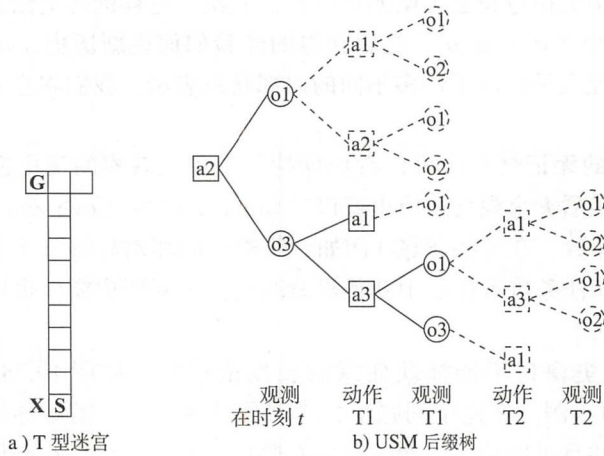


图 12.7 a) 表示一个长期依赖的 T 型迷宫 [Bakker, 2002]; b) 一个 USM 算法 [McCallum, 1995] 用到的后缀树，其中的边缘节点都用虚线进行标识

12.5 近期研究趋势

作为总结，我们将讨论近来越来越受欢迎的一些类型的方法。

本章讨论的大多数基于模型的方法都是离线技术，可以先验地确定学习器可能遇到的每种情况下所采取的动作。另一方面，在线方法只计算当前采取的动作 [Ross et al, 2008b]。专注于当前决策可以在某些领域显著地降低计算代价，因为学习器不需要规划从未遇到的状态空间的区域。然而，由于动作选择必须在单位时间步长内完成，这意味着在线搜索时间受到了严格的限制。基于离线点的方法可用于计算粗略价值函数，作为在线搜索的启发式初始化。类似地，蒙特卡罗方法对求解大型 POMDP 问题来说很具有吸引力，因为蒙特卡罗方法只需要一个生成模型（黑匣子模拟器）就能工作，并且这种方法有可能缓解维数灾难 [Thrun, 2000; Kearns et al, 2000; Silver and Veness, 2010]。

如第 11 章详细讨论的，贝叶斯强化学习技术对于求解 POMDP 问题来说是可行的，因为它们提供了一种探索和利用模型的综合方法。另外，贝叶斯强化学习技术不需要将模型学习阶段（例如，使用 Baum-Welch [Koenig and Simmons, 1996] 或其他方法 [Shani et al, 2005]）与模型开发阶段交错使用，相反它将基于模型的方法应用于未知的 POMDP，虽然这种方法看起来很朴素。[Poupart and Vlassis, 2008] 将 BEETLE 算法 [Poupart et al, 2006] 扩展为用于部分可观察 MDP 的贝叶斯强化学习（RL）方法。和其他贝叶斯 RL 方法类似，该模型由 Dirichlet 分布表示，学习涉及更新 Dirichlet 的超参（hyperparameter）。这项工作比 [Jaulmes et al, 2005] 早些时候的工作更为泛化，但该方法要求必须有一个能让学习器查询到其真实状态的如同“神谕”（oracle）一般的系统。[Ross et al, 2008a] 提出了贝叶斯自适应 POMDP 模型，这是贝叶斯强化学习的替代模型，其扩展了贝叶斯自适应的 MDP 模型 [Duff, 2002]。所有这些方法都假设状态、观测以及动作空间的大小是已知的。

策略梯度方法在参数化策略的空间中搜索, 通过在参数空间中执行梯度上升来优化策略 [Peters and Bagnell, 2010]。由于这些方法不需要估计信念状态 [Aberdeen and Baxter, 2002], 它们已经能够很容易地应用于 POMDP 问题, 且其效果也十分突出 [Peters and Schaal, 2008]。

最后, 最近的趋势是将基于模型的 RL 问题作为概率推论之一, 例如使用期望最大化来计算 MDP 中的最优策略。[Vlassis and Toussaint, 2009] 展示了如何将这些方法扩展到无模型 POMDP 的情形。一般来说, 推理方法可以为众所周知的强化学习算法提供新的思路。

408

致谢。这项工作是由 Fundac 通过 PIDDAC 计划资金资助的, 并得到项目 PTDC / EEA-ACR /73266/2006 的支持。

参考文献

- Aberdeen, D., Baxter, J.: Scaling internal-state policy-gradient methods for POMDPs. In: International Conference on Machine Learning (2002)
- Åström, K.J.: Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications* 10(1), 174–205 (1965)
- Bagnell, J.A., Kakade, S., Ng, A.Y., Schneider, J.: Policy search by dynamic programming. In: *Advances in Neural Information Processing Systems*, vol. 16. MIT Press (2004)
- Baird, L., Moore, A.: Gradient descent for general reinforcement learning. In: *Advances in Neural Information Processing Systems*, vol. 11. MIT Press (1999)
- Bakker, B.: Reinforcement learning with long short-term memory. In: *Advances in Neural Information Processing Systems*, vol. 14. MIT Press (2002)
- Baxter, J., Bartlett, P.L.: Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research* 15, 319–350 (2001)
- Baxter, J., Bartlett, P.L., Weaver, L.: Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research* 15, 351–381 (2001)
- Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research* 27(4), 819–840 (2002)
- Bonet, B.: An epsilon-optimal grid-based algorithm for partially observable Markov decision processes. In: International Conference on Machine Learning (2002)
- Boutilier, C., Poole, D.: Computing optimal policies for partially observable decision processes using compact representations. In: *Proc. of the National Conference on Artificial Intelligence* (1996)
- Brafman, R.I.: A heuristic variable grid solution method for POMDPs. In: *Proc. of the National Conference on Artificial Intelligence* (1997)
- Braziunas, D., Boutilier, C.: Stochastic local search for POMDP controllers. In: *Proc. of the National Conference on Artificial Intelligence* (2004)
- Brunskill, E., Kaelbling, L., Lozano-Perez, T., Roy, N.: Continuous-state POMDPs with hybrid dynamics. In: *Proc. of the Int. Symposium on Artificial Intelligence and Mathematics* (2008)
- Cassandra, A.R.: Exact and approximate algorithms for partially observable Markov decision processes. PhD thesis, Brown University (1998)
- Cassandra, A.R., Kaelbling, L.P., Littman, M.L.: Acting optimally in partially observable stochastic domains. In: *Proc. of the National Conference on Artificial Intelligence* (1994)
- Cassandra, A.R., Kaelbling, L.P., Kurien, J.A.: Acting under uncertainty: Discrete Bayesian models for mobile robot navigation. In: *Proc. of International Conference on Intelligent Robots and Systems* (1996)
- Cassandra, A.R., Littman, M.L., Zhang, N.L.: Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In: *Proc. of Uncertainty in Artificial Intelligence* (1997)

- Cheng, H.T.: Algorithms for partially observable Markov decision processes. PhD thesis, University of British Columbia (1988)
- Doshi, F., Roy, N.: The permutable POMDP: fast solutions to POMDPs for preference elicitation. In: Proc. of Int. Conference on Autonomous Agents and Multi Agent Systems (2008)
- Drake, A.W.: Observation of a Markov process through a noisy channel. Sc.D. thesis, Massachusetts Institute of Technology (1962)
- Duff, M.: Optimal learning: Computational procedures for Bayes-adaptive Markov decision processes. PhD thesis, University of Massachusetts, Amherst (2002)
- Dynkin, E.B.: Controlled random sequences. *Theory of Probability and its Applications* 10(1), 1–14 (1965)
- Ellis, J.H., Jiang, M., Corotis, R.: Inspection, maintenance, and repair with partial observability. *Journal of Infrastructure Systems* 1(2), 92–99 (1995)
- Feng, Z., Zilberstein, S.: Region-based incremental pruning for POMDPs. In: Proc. of Uncertainty in Artificial Intelligence (2004)
- Foka, A., Trahanias, P.: Real-time hierarchical POMDPs for autonomous robot navigation. *Robotics and Autonomous Systems* 55(7), 561–571 (2007)
- Fox, D., Burgard, W., Thrun, S.: Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research* 11, 391–427 (1999)
- Haight, R.G., Polasky, S.: Optimal control of an invasive species with imperfect information about the level of infestation. *Resource and Energy Economics* (2010) (in Press, Corrected Proof)
- Hansen, E.A.: Finite-memory control of partially observable systems. PhD thesis, University of Massachusetts, Amherst (1998a)
- Hansen, E.A.: Solving POMDPs by searching in policy space. In: Proc. of Uncertainty in Artificial Intelligence (1998b)
- Hansen, E.A., Feng, Z.: Dynamic programming for POMDPs using a factored state representation. In: Int. Conf. on Artificial Intelligence Planning and Scheduling (2000)
- Hauskrecht, M.: Value function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research* 13, 33–95 (2000)
- Hauskrecht, M., Fraser, H.: Planning treatment of ischemic heart disease with partially observable Markov decision processes. *Artificial Intelligence in Medicine* 18, 221–244 (2000)
- Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Computation* 9(8), 1735–1780 (1997)
- Hoey, J., Little, J.J.: Value-directed human behavior analysis from video using partially observable Markov decision processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(7), 1–15 (2007)
- Hoey, J., Poupart, P.: Solving POMDPs with continuous or large discrete observation spaces. In: Proc. Int. Joint Conf. on Artificial Intelligence (2005)
- Hsiao, K., Kaelbling, L., Lozano-Perez, T.: Grasping pomdps. In: Proc. of the IEEE Int. Conf. on Robotics and Automation, pp. 4685–4692 (2007)
- Jaakkola, T., Singh, S.P., Jordan, M.I.: Reinforcement learning algorithm for partially observable Markov decision problems. In: *Advances in Neural Information Processing Systems*, vol. 7 (1995)
- Jaulmes, R., Pineau, J., Precup, D.: Active Learning in Partially Observable Markov Decision Processes. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) *ECML 2005. LNCS (LNAI)*, vol. 3720, pp. 601–608. Springer, Heidelberg (2005)
- Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101, 99–134 (1998)
- Kearns, M., Mansour, Y., Ng, A.Y.: Approximate planning in large POMDPs via reusable trajectories. In: *Advances in Neural Information Processing Systems*, vol. 12. MIT Press (2000)
- Koenig, S., Simmons, R.: Unsupervised learning of probabilistic models for robot navigation. In: Proc. of the IEEE Int. Conf. on Robotics and Automation (1996)
- Kurniawati, H., Hsu, D., Lee, W.: SARSOP: Efficient point-based POMDP planning by ap-

- proximating optimally reachable belief spaces. In: *Robotics: Science and Systems* (2008)
- Lin, L., Mitchell, T.: Memory approaches to reinforcement learning in non-Markovian domains. Tech. rep., Carnegie Mellon University, Pittsburgh, PA, USA (1992)
- Lin, Z.Z., Bean, J.C., White, C.C.: A hybrid genetic/optimization algorithm for finite horizon, partially observed Markov decision processes. *INFORMS Journal on Computing* 16(1), 27–38 (2004)
- Littman, M.L.: Memoryless policies: theoretical limitations and practical results. In: *Proc. of the 3rd Int. Conf. on Simulation of Adaptive Behavior: from Animals to Animats 3*, pp. 238–245. MIT Press, Cambridge (1994)
- Littman, M.L.: Algorithms for sequential decision making. PhD thesis, Brown University (1996)
- Littman, M.L., Cassandra, A.R., Kaelbling, L.P.: Learning policies for partially observable environments: Scaling up. In: *International Conference on Machine Learning* (1995)
- Littman, M.L., Sutton, R.S., Singh, S.: Predictive representations of state. In: *Advances in Neural Information Processing Systems*, vol. 14. MIT Press (2002)
- Loch, J., Singh, S.: Using eligibility traces to find the best memoryless policy in partially observable Markov decision processes. In: *International Conference on Machine Learning* (1998)
- Lovejoy, W.S.: Computationally feasible bounds for partially observed Markov decision processes. *Operations Research* 39(1), 162–175 (1991)
- Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence* 147(1-2), 5–34 (2003)
- McCallum, R.A.: Overcoming incomplete perception with utile distinction memory. In: *International Conference on Machine Learning* (1993)
- McCallum, R.A.: Instance-based utile distinctions for reinforcement learning with hidden state. In: *International Conference on Machine Learning* (1995)
- McCallum, R.A.: Reinforcement learning with selective perception and hidden state. PhD thesis, University of Rochester (1996)
- Meuleau, N., Kim, K.E., Kaelbling, L.P., Cassandra, A.R.: Solving POMDPs by searching the space of finite policies. In: *Proc. of Uncertainty in Artificial Intelligence* (1999a)
- Meuleau, N., Peshkin, L., Kim, K.E., Kaelbling, L.P.: Learning finite-state controllers for partially observable environments. In: *Proc. of Uncertainty in Artificial Intelligence* (1999b)
- Monahan, G.E.: A survey of partially observable Markov decision processes: theory, models and algorithms. *Management Science* 28(1) (1982)
- Ng, A.Y., Jordan, M.: PEGASUS: A policy search method for large MDPs and POMDPs. In: *Proc. of Uncertainty in Artificial Intelligence* (2000)
- Oliehoek, F.A., Spaan, M.T.J., Vlassis, N.: Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research* 32, 289–353 (2008)
- Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of Markov decision processes. *Mathematics of Operations Research* 12(3), 441–450 (1987)
- Parr, R., Russell, S.: Approximating optimal policies for partially observable stochastic domains. In: *Proc. Int. Joint Conf. on Artificial Intelligence* (1995)
- Peters, J., Bagnell, J.A.D.: Policy gradient methods. In: *Springer Encyclopedia of Machine Learning*. Springer, Heidelberg (2010)
- Peters, J., Schaal, S.: Natural actor-critic. *Neurocomputing* 71, 1180–1190 (2008)
- Pineau, J., Thrun, S.: An integrated approach to hierarchy and abstraction for POMDPs. Tech. Rep. CMU-RI-TR-02-21, Robotics Institute, Carnegie Mellon University (2002)
- Pineau, J., Gordon, G., Thrun, S.: Point-based value iteration: An anytime algorithm for POMDPs. In: *Proc. Int. Joint Conf. on Artificial Intelligence* (2003)
- Platzman, L.K.: A feasible computational approach to infinite-horizon partially-observed Markov decision problems. Tech. Rep. J-81-2, School of Industrial and Systems Engineering, Georgia Institute of Technology, reprinted in working notes AAAI, Fall Symposium on Planning with POMDPs (1981)
- Poon, K.M.: A fast heuristic algorithm for decision-theoretic planning. Master's thesis, The Hong-Kong University of Science and Technology (2001)
- Porta, J.M., Spaan, M.T.J., Vlassis, N.: Robot planning in partially observable continuous

- domains. In: *Robotics: Science and Systems* (2005)
- Porta, J.M., Vlassis, N., Spaan, M.T.J., Poupart, P.: Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research* 7, 2329–2367 (2006)
- Poupart, P.: Exploiting structure to efficiently solve large scale partially observable Markov decision processes. PhD thesis, University of Toronto (2005)
- Poupart, P., Boutilier, C.: Bounded finite state controllers. In: *Advances in Neural Information Processing Systems*, vol. 16. MIT Press (2004)
- Poupart, P., Vlassis, N.: Model-based Bayesian reinforcement learning in partially observable domains. In: *International Symposium on Artificial Intelligence and Mathematics, ISAIM* (2008)
- Poupart, P., Vlassis, N., Hoey, J., Regan, K.: An analytic solution to discrete Bayesian reinforcement learning. In: *International Conference on Machine Learning* (2006)
- Ross, S., Chaib-draa, B., Pineau, J.: Bayes-adaptive POMDPs. In: *Advances in Neural Information Processing Systems*, vol. 20, pp. 1225–1232. MIT Press (2008a)
- Ross, S., Pineau, J., Paquet, S., Chaib-draa, B.: Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research* 32, 664–704 (2008b)
- Roy, N., Gordon, G.: Exponential family PCA for belief compression in POMDPs. In: *Advances in Neural Information Processing Systems*, vol. 15. MIT Press (2003)
- Roy, N., Thrun, S.: Coastal navigation with mobile robots. In: *Advances in Neural Information Processing Systems*, vol. 12. MIT Press (2000)
- Roy, N., Gordon, G., Thrun, S.: Finding approximate POMDP solutions through belief compression. *Journal of Artificial Intelligence Research* 23, 1–40 (2005)
- Sanner, S., Kersting, K.: Symbolic dynamic programming for first-order POMDPs. In: *Proc. of the National Conference on Artificial Intelligence* (2010)
- Satia, J.K., Lave, R.E.: Markovian decision processes with probabilistic observation of states. *Management Science* 20(1), 1–13 (1973)
- Seuken, S., Zilberstein, S.: Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems* (2008)
- Shani, G., Brafman, R.I.: Resolving perceptual aliasing in the presence of noisy sensors. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) *Advances in Neural Information Processing Systems*, vol. 17, pp. 1249–1256. MIT Press, Cambridge (2005)
- Shani, G., Brafman, R.I., Shimony, S.E.: Model-Based Online Learning of POMDPs. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) *ECML 2005. LNCS (LNAI)*, vol. 3720, pp. 353–364. Springer, Heidelberg (2005)
- Shani, G., Brafman, R.I., Shimony, S.E.: Forward search value iteration for POMDPs. In: *Proc. Int. Joint Conf. on Artificial Intelligence* (2007)
- Shani, G., Poupart, P., Brafman, R.I., Shimony, S.E.: Efficient ADD operations for point-based algorithms. In: *Int. Conf. on Automated Planning and Scheduling* (2008)
- Silver, D., Veness, J.: Monte-carlo planning in large POMDPs. In: Lafferty, J., Williams, C.K.I., Shawe-Taylor, J., Zemel, R., Culotta, A. (eds.) *Advances in Neural Information Processing Systems*, vol. 23, pp. 2164–2172 (2010)
- Simmons, R., Koenig, S.: Probabilistic robot navigation in partially observable environments. In: *Proc. Int. Joint Conf. on Artificial Intelligence* (1995)
- Singh, S., Jaakkola, T., Jordan, M.: Learning without state-estimation in partially observable Markovian decision processes. In: *International Conference on Machine Learning* (1994)
- Singh, S., James, M.R., Rudary, M.R.: Predictive state representations: A new theory for modeling dynamical systems. In: *Proc. of Uncertainty in Artificial Intelligence* (2004)
- Smallwood, R.D., Sondik, E.J.: The optimal control of partially observable Markov decision processes over a finite horizon. *Operations Research* 21, 1071–1088 (1973)
- Smith, T., Simmons, R.: Heuristic search value iteration for POMDPs. In: *Proc. of Uncertainty in Artificial Intelligence* (2004)
- Smith, T., Simmons, R.: Point-based POMDP algorithms: Improved analysis and implementation. In: *Proc. of Uncertainty in Artificial Intelligence* (2005)
- Sondik, E.J.: The optimal control of partially observable Markov processes. PhD thesis, Stanford University (1971)
- Spaan, M.T.J., Vlassis, N.: A point-based POMDP algorithm for robot planning. In: *Proc. of*

- the IEEE Int. Conf. on Robotics and Automation (2004)
- Spaan, M.T.J., Vlassis, N.: Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research* 24, 195–220 (2005a)
- Spaan, M.T.J., Vlassis, N.: Planning with continuous actions in partially observable environments. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation* (2005b)
- Spaan, M.T.J., Veiga, T.S., Lima, P.U.: Active cooperative perception in network robot systems using POMDPs. In: *Proc. of International Conference on Intelligent Robots and Systems* (2010)
- Sridharan, M., Wyatt, J., Dearden, R.: Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs. *Artificial Intelligence* 174, 704–725 (2010)
- Stankiewicz, B., Cassandra, A., McCabe, M., Weathers, W.: Development and evaluation of a Bayesian low-vision navigation aid. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 37(6), 970–983 (2007)
- Stratonovich, R.L.: Conditional Markov processes. *Theory of Probability and Its Applications* 5(2), 156–178 (1960)
- Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press (1998)
- Theocharous, G., Mahadevan, S.: Approximate planning with hierarchical partially observable Markov decision processes for robot navigation. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation* (2002)
- Thrun, S.: *Monte Carlo POMDPs*. In: *Advances in Neural Information Processing Systems*, vol. 12. MIT Press (2000)
- Thrun, S., Burgard, W., Fox, D.: *Probabilistic Robotics*. MIT Press (2005)
- Varakantham, P., Maheswaran, R., Tambe, M.: Exploiting belief bounds: Practical POMDPs for personal assistant agents. In: *Proc. of Int. Conference on Autonomous Agents and Multi Agent Systems* (2005)
- Vlassis, N., Toussaint, M.: Model-free reinforcement learning as mixture learning. In: *International Conference on Machine Learning*, pp. 1081–1088. ACM (2009)
- Wang, C., Khordon, R.: Relational partially observable MDPs. In: *Proc. of the National Conference on Artificial Intelligence* (2010)
- White, C.C.: Partially observed Markov decision processes: a survey. *Annals of Operations Research* 32 (1991)
- Wiering, M., Schmidhuber, J.: HQ-learning. *Adaptive Behavior* 6(2), 219–246 (1997)
- Wierstra, D., Wiering, M.: Utile distinction hidden Markov models. In: *International Conference on Machine Learning* (2004)
- Williams, J.D., Young, S.: Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language* 21(2), 393–422 (2007)
- Williams, J.K., Singh, S.: Experimental results on learning stochastic memoryless policies for partially observable Markov decision processes. In: *Advances in Neural Information Processing Systems*, vol. 11 (1999)
- Zhang, N.L., Liu, W.: Planning in stochastic domains: problem characteristics and approximations. Tech. Rep. HKUST-CS96-31, Department of Computer Science, The Hong Kong University of Science and Technology (1996)
- Zhou, R., Hansen, E.A.: An improved grid-based approximation algorithm for POMDPs. In: *Proc. Int. Joint Conf. on Artificial Intelligence* (2001)

预测性定义状态表示

David Wingate

摘要

状态是动态系统的核心概念。对于任何时间序列问题，如滤波、规划或预测，算法模型将过去的重要信息概括成某种状态变量。在本章中，我们先回顾状态的概念，重点探讨同一动态系统的多种不同状态表示具有的不同的理论和计算性质。之后，我们重点关注预测性定义的状态表示模型，这些模型将未来短期内的统计数据表示为状态，而与之相对的经典方法将状态描述为潜在的、不可取样的量。换言之，在预测性模型中过去的信息被总结成未来短期内的动作及可观测的预测，用于进一步预测无限远的未来。这个表示思想能应用于任何动态系统问题，而且它有利于基于模型的强化学习问题。因为在基于模型的 RL 中学习器必须在线学习状态表示及动态模型，而预测性模型仅使用可取样的统计量来定义状态，所以预测性表示的学习算法往往是直截了当且有吸引力的。本章研究预测性定义状态表示的概念、重要的附属结构（如系统动态矩阵等）以及模型的表达能力和学习能力。

一部分该章内容最初发布在《Exponential Family Predictive Representations of State》

415 [David Wingate, 2008]。

13.1 简介

本章考虑预测性定义状态表示的动态系统模型。为引出这个模型，我们首先审视状态的基本概念，并考虑为什么状态对学习器很重要，为什么不同的表示有不同的理论或计算性质，特别是当系统在线学习状态表示和系统动态模型时差异更明显。

例如，在基于模型的强化学习初始化过程中，学习器必须通过与环境交互来获取数据从而学习模型。这些模型与规划算法一起使用，以帮助学习器推理未来并选择最佳动作。因此基本的可行模型必须能够根据对未来的观测和奖励预测动作。为了实现最优预测，模型必须能够概括过去的重要经验。这种对过去的总结称为状态，稍后我们将正式地定义这个概念。现在，我们先将状态非正式地定义为学习器对环境中事务状态的知识总结。例如，在马尔可夫决策过程（MDP）中，最近的取样确定状态，意味着模型的构建仅涉及学习状态之间如何转换。然而，在部分可观察系统中，最近的取样不构成状态，因此，学习器必须学习哪些历史信息是核心的，如何记住它，以及如何使用它来预测未来。

本章深入探讨了一种称为“预测性定义状态表示”的特定状态表示。在预测性定义状态表示模型中，学习器将之前的信息概括为关于未来短期内的一组预测，并能用于学习器对无限远未来做进一步预测。这种状态表示似乎违反直觉，但其实想法很简单：每个状态都是之前信息的总结，每个之前信息都暗示着未来的分布——那么，为什么不直接用这种分布的统计数据来代表状态呢？

在这里，我们将探讨这种表示的理论基础，并将它们的计算和表示特征与传统状态概念的特征进行比较。我们通过考虑模型构建的两个关键部分——学习状态表示和学习算法（该算法允许学习器在对新信息的响应中准确地保持状态），关注模型如何直接从在线获取的数据中学习。

13.1.1 状态是什么

本章主要关注状态的概念以及该如何表示它。现在，我们更精确地定义状态，看点放在状态的多个可接受的不同表示上。在这里，我们是从学习器的视角讨论状态，与之相对的是全知视角——该视角关注必须学习模型的学习器。 [416]

什么是“状态”？通俗地说，状态是学习器发现的自身当前所处情况：对于机器人，状态可以是其所有执行器的位置和角度，以及其地图坐标、电池状态、其试图实现的目标等。在马尔可夫系统中，所有这些变量都被赋予学习器。然而，在部分可观察系统中，学习器可能不能立即访问关于其情况的所有信息：传感器损坏的机器人可能不知道其手臂的确切位置；股票交易者可能不知道他投资的所有公司的长期战略是什么；棒球击球员可能不知道棒球接近时有多快。

在部分可观察的域中，对状态有一个更加正式的定义：状态是学习器可用于发现其当前状况的所有信息的总结。这些信息包含在它所经历动作和取样的历史中，这意味着完整的历史构成完美的状态，这也是衡量一个状态表示的标准。当然，学习器通常想要简化这个历史，而不丢失信息！否则它将存储越来越大量的信息，因为它与环境交互的时间越来越长。由此我们对状态的正式定义为：

状态是足以预测未来奖励和取样分布的任何有限维度的历史的统计。

有时我们简化这个定义为“状态是历史的充分统计数据”。

以这种方式定义状态意味着状态拥有条件独立的性质：未来取样的分布与当前状态有关，而与过去信息条件独立，即

$$p(\text{future}|\text{state}, \text{past}) = p(\text{future}|\text{state})$$

这意味着状态已经包含了与预测未来有关的所有历史信息，因此我们可以丢弃历史本身。事实证明，学习器可以根据这个历史信息的总结而执行最佳动作 [Astrom, 1965]。因此表示状态、维持状态、总结历史、预测未来及选择最佳动作之间存在着密切联系。

不幸的是，“状态”这个词存在一些相互矛盾的使用。例如，在 POMDP 中，模型假定基础“状态”代表了过程的“真实”状态，这是学习器不可取样的（必要的话，我们将这些基本状态称为“潜在状态”）。在 POMDP 术语中，学习器总结具有“可信状态”的历史，其是在潜在状态上的分布。根据我们的定义，可信状态是历史的充分统计数据。然而，潜在状态不满足数据充分性。正如我们将看到的，本章的贡献之一是引入了许多不涉及任何潜在状态的状态概念。 [417]

13.1.2 哪一个状态表示

实际中存在许多令人满意的对历史的概括。为了说明这一点，考虑机器人定位问题的示例。一个小机器人漫步穿过建筑物，需要跟踪其位置，但它只有一个相机传感器。机器人的位置有许多可能的表示。例如，可以根据 x 、 y 坐标上的分布来捕获其姿态。然而，也可以根据极坐标上的分布来描述。笛卡儿坐标和极坐标是状态的不同表示，它们具有相同的表

达性,但两者都是学习器内部的。从环境的角度来看,两者都不够准确,或更正确、更有用——笛卡儿坐标或极坐标不能被准确地保持,只能给出相机图像。易知有无数个这样的状态表示:状态的任何一对一变换后仍然是状态,并且将冗余信息添加到状态后仍然是状态。

由于存在多个令人满意的状态表示,那么在所有可能的状态概念中,应该优选哪一个呢?有许多标准可用于比较不同的状态表示。例如,如果一个状态表示满足以下条件,可认为它是较好的:

- 它易于程序员理解及使用。
- 它以某种方式匹配另一学习器的状态概念。
- 它具有良好的计算或统计特性。

不是每个历史统计都足以预测未来,这意味着一些表示可能只构成近似状态。在近似足够的表示中,如果满足以下条件可能被优选:

- 它比其他的表示表达得更多。
- 它表示的比其他的少,但是足够我们完成想要完成的事情(例如,最佳控制系统)。

因此,即使具有同样表达性的状态表示,也存在优先级。

因为我们对学习学习器感兴趣,所以我们关注可学习(存在有效的学习算法)的状态表示。一种状态表示比另一种更可学习,这引出了不同状态表示之间的基本区别:基本(grounded)^①状态表示是指其中状态的每个分量仅使用统计量(可能是在未来或过去的任何取样值)来定义,潜在状态表示包括不可观测量。在机器人定位示例中,笛卡儿坐标和极坐标都是状态的潜在表示,因为机器人没有明确的取样值;只有根据相机图像特征定义的状态表示可以称为基本状态表示。

418

我们将进一步对基本状态表示类别进行区分。一些基本表示可以根据过去的取样来定义,如 k 阶马尔可夫模型(其中过去 k 次取样构成状态),还有一些可以根据当前取样来定义。

本章的主题是第三类基本表示形式——预测性定义状态表示。在预测定义状态表示中,将与未来取样特征有关的统计量表示为状态。这些统计量是灵活的:它们可代表未来短期内的分布参数,代表未来随机变量的期望,代表特定未来动作和取样的密度,或代表基于未来动作的取样值的声明等。我们将在这一章中研究这种状态表示,以及学习和维持这种状态的算法。

13.1.3 为什么使用预测性定义模型

为什么要研究预测性定义状态表示模型?我们的动机有三个原因:

1) 可学习性。本章讨论的核心问题是从数据中学习动态系统的模型。预测性定义模型的所有参数都与可取样量有直接的、统计的关系,这一事实表明预测性定义模型可能比经典模型更易学习。例如,预测性线性高斯(PLG)模型(稍后讨论的卡尔曼滤波器的预测状态表示版本[Rudary et al,2005])的参数估计算法已证明在统计上是一致的,这是一个很强的学习保证。

2) 表征能力。迄今为止,大多数具有预测状态的模型已证明和它们的经典模型相比具有至少同等的表达性。例如,PLG与卡尔曼滤波器[Rudary et al, 2005]具有一样的表达

① 一些学科对 grounded 这词有更具体和专业性的定义,我们不管它们。

性, 线性 PSR (在 13.2 节中讨论) 严格地比 POMDP 更具表达性 [James, 2005]。也就是说, 有一些域可以用有限的 PSR 建模, 而不能用任何有限的 POMDP 建模, 但是每个有限的 POMDP 都可以通过有限的 PSR 来建模。这种表征能力是在不牺牲紧凑性的前提下实现的: 线性 PSR 绝对比其等效的 POMDP 更紧凑, PLG 绝对比其等效的卡尔曼滤波器更紧凑, 存在 PSR 比其等价 POMDP 具有指数级更少的参数 [Littman et al, 2002]。对于给定的数据集, 具有较少参数的模型可能具有较高的统计效率, 这对于可学习性是有用的。

3) 普遍性。从函数近似和泛化的角度来看, 对未来预测的知识表示具有更大的吸引力。[419] 例如, [Rafols et al, 2005] 提供了一些初步证据, 预测性表示比潜在表示为泛化提供了更好的基础。为了直观地看到这一点, 考虑迷宫导航里分配状态的问题。使用预测定义表示法表示, 两个状态在未来的分布相似时, 两个状态彼此“接近”; 如果这是真的, 它们应该被分配相似的值。但是如果学习器使用比如笛卡尔坐标作为状态表示, 在欧几里得空间中附近的两个状态不一定具有相似的值。典型的例子是在墙两侧的两个状态: 尽管两个状态看起来是接近的, 但学习器必须通过迷宫行进长距离才能从一点到达另一点, 所以它们应该被分配不同的值, 平滑函数逼近器难以做到这点。[Littman et al, 2002] 也提出, 在组成领域中, 预测状态表示也可用于学习其他预测, 并指出在许多情况下“早期 (预测) 问题的解决方案提供了一些特征, 有利于后来的 (预测) 问题的泛化。”这也部分地在 [Izadi and Precup, 2005] 中得到证实, 其表明 PSR 具有压缩状态空间对称性的能力。

本章的其余部分调查预测性定义状态表示模型的重要概念和结果。13.2 节介绍 PSR, 13.3 节讨论如何从数据中学习 PSR 模型, 13.4 节讨论 PSR 中的规划, 13.5 节介绍 PSR 的扩展, 13.6 节调查具有预测性定义状态性质的其他模型, 最后, 13.7 节提出了一些总结性的想法。

13.2 PSR

POMDP 存在潜在状态: POMDP 首先描述潜在状态空间、这些状态之间的转变以及它们产生的取样; 它将历史的足够统计定义为对这些潜在状态的分布等。在特定情况下, 如设计者知道系统中存在潜在状态并且知道它们的关系, 该模型是方便的。然而, 有许多作者指出, 虽然 POMDP 易于描述, 但是很难从数据中学习 [Nikovski, 2002; Shatkay and Kaelbling, 2002], 这也是我们主要的关注点。

在本节中, 我们转向使用了离散取样的受控动态系统的替代模型, 即“预测状态表示” (PSR)[⊖]。PSR 由 [Littman et al, 2002] 提出, 是被首先提出的预测性定义状态表示模型之一。[420] 像 POMDP 一样, PSR 模型能够捕获具有离散动作和离散取样的动态系统。重要的区别是, POMDP 是围绕潜在状态建立的, 但是 PSR 从未提及潜在状态; 相反, PSR 将状态表示为关于未来的统计的集合。这将对学习产生积极的影响, 但正如我们能看到的, 它会损失建模能力: 我们将看到 PSR 可以对有限状态 POMDP 适用的任何系统建模, 并且许多 POMDP 规划算法直接适用于 PSR。

我们现在将介绍解释 PSR 所需的术语。

13.2.1 历史及测试

“历史”被定义为过去发生的一系列动作和取样 $a_1 o_1 a_2 o_2 \cdots a_m o_m$ 。“测试”被定义为未

⊖ 不巧的是, PSR 这个名字更适合用于表示整个类别的模型。

来动作和取样的可能序列 $a^1 o^1 a^2 o^2 \cdots a^n o^n$ 。学习器不一定采取测试中定义的动作，仅代表学习器可能希望推理的一个可能的未来。

图 13.1 说明了测试的概念。该图中有两个机器人在迷宫中。在这个问题中，动作包括“向左移动”、“向右移动”等，取样包括“相遇”、“看到粉色墙”、“看到蓝色墙”等。对左边的机器人来说，它有基于动作的不同未来的分配：如果向左移动，它会遇到粉墙；如果向右然后向上移动，它会遇到蓝墙；如果向右然后向下移动，它会遇到绿墙。右边的机器人在迷宫的另一个位置，因此对未来有不同的分配：若向左，则它会遇到黄墙而不是蓝墙。在这个简单例子中我们可以发现足够详细的一组对未来足够长的预测能够使两个位置不存在歧义。这正是 PSR 中状态表示的优点。

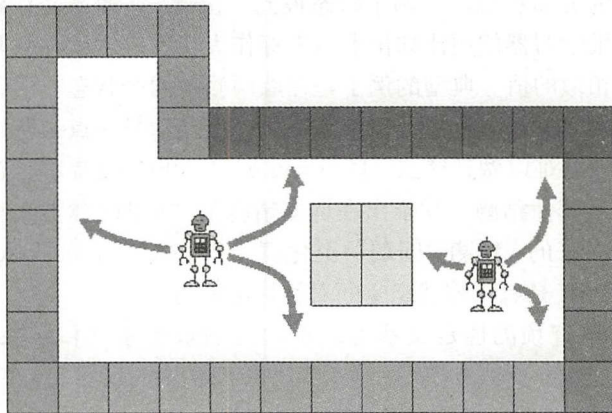


图 13.1 对未来状态的预测。潜在状态表示定义为 x, y 坐标，但预测定义状态定义未来的可能性

13.2.2 测试的预测

测试是 PSR 使用的状态表达中核心的部分。它们也是 PSR 依赖于理论结果的数学对象以及大多数学习算法的核心思想。这是因为通过使用测试可以由任何给定的历史数据得到未来动作及取样的可行序列：每一个不同可行未来对应不同的测试，并且有一定概率（可能为零）每个测试在任意历史信息中发生。

我们定义预测的测试结果为 $t = a^1 o^1 a^2 o^2 \cdots a^n o^n$ 。假定学习器从历史 $h = a_1 o_1 a_2 o_2 \cdots a_m o_m$ 开始，并且它已经完成了 h 说明的动作和得出了 h 指定的取样结果。对 t 的预测是指接下来的 n 个动作及取样是完全符合 t 的概率

$$p(t|h) = \Pr(o_{m+1} = o^1, o_{m+2} = o^2, \cdots, o_{m+n} = o^n | h, a_{m+1} = a^1, \cdots, a_{m+n} = a^n)$$

测试中的动作通过开环的方式工作，不依赖于未来的信息的反馈，所以

$$\Pr(a_n | a_1, o_1, \cdots, a_{n-1}, o_{n-1}) = \Pr(a_n | a_1, \cdots, a_n)$$

在 [Bowling et al, 2006] 中讨论了这个性质的重要性，特别是在判断测试可行性时。为了方便标记，我们采用以下速记： $T = \{t_1, t_2, \cdots, t_n\}$ 为设定的测试集， $p(T|h) = [p(t_1|h), p(t_2|h), \cdots, p(t_n|h)]^T$ 为测试 T 的预测。

13.2.3 系统动态向量

系统动态向量 [Singh et al, 2004] 是为了介绍 PSR 而引入的概念构造。这个向量描述着动态系统的过程：每个可行的测试 t 都在该向量中被预测，表示为 $p(t|\emptyset)$ ，即从空历史开始

的 t 的预测。这些测试通常按照长度词典顺序排列，从最短到最长。一般来说，这是一个无限长的向量，但从理论角度看，它仍然是有用的。我们使用 $a_i^m o_i^n$ 分别代表在 t 时刻的第 m 个动作及第 n 个取样。

$$V = \left[p(a_1^1 o_1^1 | \phi), p(a_1^1 o_1^2 | \phi), \dots, p(a_1^m o_1^n | \phi), p(a_1^1 o_1^1 a_2^1 o_2^1 | \phi), \dots \right]$$

422

系统动态向量是可以独立表示的：具有离散取样的每个动态系统（包括可控的、部分可观察的、不稳定的、可遍历的或任何其他类型的）可以完全由不涉及任何潜在状态的系统动态向量表示。

13.2.4 系统动态矩阵

系统动态矩阵 D （如图 13.2 所示）是用于分析历史属性和测试属性的重要理论结构，并且在许多 PSR 学习算法中作为核心对象。

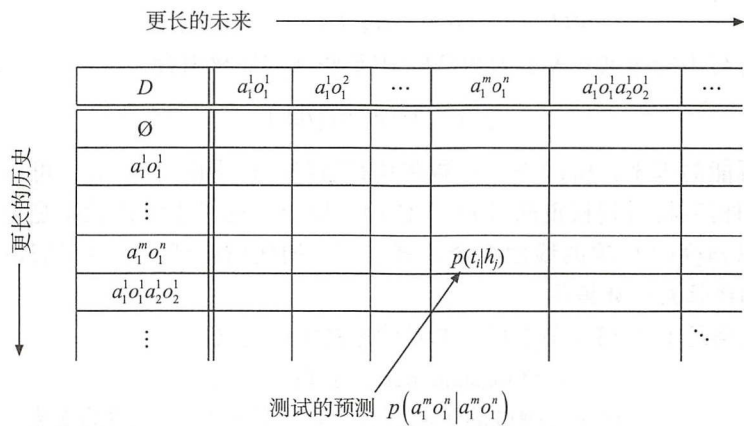


图 13.2 系统动态矩阵

系统动态矩阵是通过在所有可行历史上调节系统动态向量 V 来获得的。在该矩阵中，第一行是系统动态向量，对应于来自空历史的每个测试的预测。每个可能的历史在矩阵中具有一行，而这行中的项通过调节该历史上的系统动态向量来获得。矩阵中的 (i, j) 项是第 j 个测试根据第 i 个历史的预测：

$$D_{ij} = p(t_j | h_i) = \frac{p(h_i t_j | \phi)}{p(h_i | \phi)}$$

423

注意，系统动态向量具有计算系统动态矩阵所需的全部信息。测试和历史以长度透视法排列，长度不断增加。矩阵的行和列是无限的，像系统动态向量一样，它是一个动态系统的完整描述。

13.2.5 充分的数据集

系统动态矩阵本质上定义了一个充分统计的概念，并提出了几种可能的学习算法和状态更新机制。即使系统动态矩阵具有无限数量的行和列，但只要它具有有限秩，则一定存在至少一组有限的线性无关列对应于一组线性无关测试。我们称这些与线性无关列相关的测试为核心测试。类似地，一定存在对应于线性无关历史的线性无关行。我们将与这些线性无关行

相关的历史称为核心历史。

实际上, 系统动态矩阵的有限秩在特定条件下是存在的, 例如 POMDP, 如 13.2.10 节所述。我们将看到估计和利用系统动态矩阵的秩是许多 PSR 学习算法的关键组成部分。

13.2.6 状态

我们现在准备讨论 PSR 的核心思想: PSR 将核心测试的预测描述为状态, 即未来可能动作及其导致的未来可能取样的概率。核心测试是 PSR 的核心思想, 因为根据定义, 其他每一列 (即关于未来可能的任何预测) 都可以计算为这些核心测试列的加权组合。重要的是, 权重独立于时间和历史, 这意味着它们仅需要估计一次就可以用于学习器的整个运算。

要了解一组核心测试的预测如何构成状态, 请考虑一个特定的历史 h_t 。假设学习器知道哪些测试是核心测试。我们将这些核心测试称为集合 Q , 并且进一步假设该学习器可以访问包含历史 h_t 预测的向量:

$$p(Q|h_t) = [p(q_1|h_t), p(q_2|h_t), \dots, p(q_n|h_t)]$$

可以将对应于 h_t 行中的每列 c 表示为 $p(Q|h_t)$ 中某些项的加权组合:

$$p(c|h_t) = m_c^T p(Q|h_t)$$

因为列对应于可能的未来, 所以该学习器在具有适当的权重向量 m_c 后, 可以预测任何它需要的未来。我们稍后将看到权重向量 m_c 不依赖于历史, 这对于维持状态是至关重要的。因为学习器可以以 $p(Q|h_t)$ 中项的线性组合预测它需要的任何测试, 所以我们说这些核心测试的预测是系统的线性充分数据集。

424

因此, 我们满足了在 13.1 节中概述的对状态的关键定义:

$$p(\text{future}|\text{state}, \text{past}) = p(\text{future}|\text{state})$$

在这种情况下, 一旦我们有核心测试的预测, 就可以计算任何感兴趣的未来, 并且可以丢弃历史。

13.2.7 更新状态

我们已经表明, 如果给出一组预测, 可以计算不同未来动作-取样的概率。但是, 我们如何通过这些预测响应新的动作和取样呢?

随机性的定义测试能帮我们解决这个问题。假设我们有一组核心测试 Q 及其在当前历史 h 的预测 $p(Q|h)$, 我们采取新的动作 a , 并得到了一个新的取样 o 。为了更新状态, 我们要计算 $p(Q|hao)$, 即新历史 hao 下的状态 (hao 是 h 序列尾加上新动作 a 及新取样 o)。我们使用 $p(Q|h)$ 来计算 $p(Q|hao)$ 。根据定义, 核心测试的预测 $q_i \in Q$ 的更新是:

$$p(q_i|hao) = \frac{p(aoq_i|h)}{p(ao|h)}$$

aoq_i 表示测试包含动作 a 、取样 o 以及 q_i 所包含的动作和取样。注意, 右侧的量是严格地根据历史 h 的预测来定义的, 而且由于 $p(Q|h)$ 可以根据 h 预测所有的未来, 我们可以仅通过计算一步测试 ao 的预测及对核心测试的一步扩展 aoq_i 来更新状态。

该公式适用于所有 PSR, 无论它们使用线性的充分数据集还是非线性的充分数据集。如下面所讨论的, 在线性充分数据集的情况下, 状态更新采取特别方便的形式。

13.2.8 线性 PSR

线性 PSR 是建立在线性充分数据集上的。在线性 PSR 中, 存在权重向量 $m_c \in \mathbb{R}^{|Q|}$, 使得在任意历史 h 下任意测试 c 的预测值 $p(c|h) = m_c^\top p(Q|h)$ 。这意味着更新单个核心测试 $q_i \in Q$ 的预测值可以有效地以闭环形式完成。从历史 h 、采取动作 a 和观察取样 o 后:

$$p(q_i|hao) = \frac{p(aoq_i|h)}{p(ao|h)} = \frac{m_{aoq_i}^\top p(Q|h)}{m_{ao}^\top p(Q|h)} \quad (13.1)$$

425

这个公式显示了如何以方便的闭环形式递归地更新单个测试。以前, 我们说, 给定某一历史 h 下的一组核心测试的预测, 相同行中的任何其他列可以计算为 $p(Q|h)$ 的加权和。在这里, 我们看到, 为了更新状态, 只需要两个预测: 一步测试 $p(ao|h)$ 的预测和一步扩展 $p(aoq_i|h)$ 的预测。因此, 学习器仅需要知道作为一步测试的权重向量 m_{ao} 和作为一步扩展的权重的 m_{aoq_i} 。我们可以通过定义矩阵 M_{ao} 来将所有核心测试的更新合并为单个更新, 其中第 i 行等于 $m_{aoq_i}^\top$ 。同时更新所有核心测试等效于归一化矩阵向量相乘:

$$p(Q|hao) = \frac{M_{ao} p(Q|h)}{m_{ao}^\top p(Q|h)} \quad (13.2)$$

这使得我们可以重复地更新状态。

学习器不需要为它希望预测的系统中的每个可能的测试学习一个权重向量。如果它已经学习了一步测试和一步扩展的权重, 那么足以任何测试创建一个预测。这是通过将模型向前滚动到未来而实现的。对任意一个测试 $t = a^1 o^1 \cdots a^n o^n$, 其预测可以通过下式计算:

$$p(t|h) = m_{a^n o^n}^\top M_{a^{n-1} o^{n-1}} \cdots M_{a^1 o^1} p(Q|h) \quad (13.3)$$

这可以通过考虑系统动态矩阵 [Singh et al, 2004] 或通过考虑等效 POMDP [Littman et al, 2002] 的参数来导出。

13.2.9 线性 PSR 与 POMDP 的关联

通过线性 PSR 和 POMDP 之间的映射, 可以验证线性 PSR 的许多理论性质。在这里, 我们将研究一种关系, 为它们的一般关系提供一些参考: 我们表明两个模型都有线性更新和预测方程, 线性 PSR 的参数是等效 POMDP 参数的相似变换。

在本节中, 我们只考虑有 n 个状态的有限 POMDP。我们使用 $O^{a,o}$ 来表示取样矩阵, 其是 $n \times n$ 的矩阵。 $O_{i,i}^{a,o}$ 表示当采取动作 a 时在状态 i 中取样为 o 的概率。 T^a 是 $n \times n$ 的转换矩阵, 其列表示为 $T(s|s,a)$ 。

更新 POMDP 中的状态类似于更新 PSR 中的状态。给定可信状态 b_h 和新动作 a 和观察 o , 则可信状态 b_{hao} 为:

$$b_{hao} = \frac{O^{a,o} T^a b_h}{\mathbf{1}^T O^{a,o} T^a b_h}$$

426

其中 $\mathbf{1}$ 是 $n \times 1$ 的单位向量。注意与公式 (13.2) 中的 PSR 状态更新的强相似性: 可以离线计算一次矩阵乘积 $O^{a,o} T^a$ 以创建单个矩阵, 这意味着两种表示中更新状态的计算复杂度是相同的。

在 POMDP 模型中, 当 $t = a^1 o^1 \cdots a^n o^n$ 时, POMDP 必须计算

$$p(t|h) = \mathbf{1}^T O^{a^n, o^n} T^{a^n} \cdots O^{a^1, o^1} T^{a^1} b_h = w_t^\top b_h$$

通过该公式我们能再次预计算向量矩阵。注意与公式 (13.3) 中 PSR 预测方程的相似性。两者都可以将任何预测作为其各自状态向量中项的加权组合, 且权重不依赖于历史的。

线性 PSR 和 POMDP 也具有密切相关的参数。为了说明这一点, 我们为 n 维 POMDP 定义结果矩阵 U 和给定的 n 核心测试 Q 。 U_{ij} 表示在状态 i 的历史下成功执行测试 j 的概率。注意, 如果设置的测试 Q 构成一组核心测试, U 将具有满秩。给定 U , 则 [Littman et al, 2002]:

$$\begin{aligned} M_{ao} &= U^{-1} T^a O^{a,o} U \\ m_{ao} &= U^{-1} T^a O^{a,o} \mathbf{1} \end{aligned}$$

换句话说, 线性 PSR 的参数与等效 POMDP 的参数是相似转换的。这个事实有利于将 POMDP 规划算法转换为 PSR 的规划算法, 这将在 13.4 节中提到。

13.2.10 线性 PSR 的理论结果

关于线性 PSR 的性质有各种理论结果。在这里, 我们探讨其中一些最重要的性质。

表现力。早期显示, 通过给出将 POMDP 转换为 PSR 的构造性证明, 每个 POMDP 都可以用一个 PSR 来表示 [Littman et al, 2002]。这个结果由 [James, 2005] 推广, 其指出: “有限 PSR 能够模拟所有有限 POMDP、HMM、MDP、MC、历史窗口框架、多样性表示、可解释的 OOM 及可解释的 IO-OOM。”实际上, PSR 比 POMDP 更具表现力。[James, 2005] 表明存在不能由任何有限 POMDP、隐马尔可夫模型、MDP、马尔可夫链、历史窗口、多样性表示、可解释的 OOM 或可解释的 IO-OOM 建模的有限 PSR。

紧凑性。PSR 与 POMDP 一样紧凑。[James, 2005] 指出, “可以由有限 POMDP (包括所有 HMM、MDP 和 MC) 建模的每个系统也可以通过有限 PSR 来建模, 使用的核心测试数量小于或等于在 POMDP 中的状态数量”。此外, PSR 捕获状态所需的核心测试的长度存在严谨的界限。[Wolfe, 2009] 表明 “对于任何秩为 n 的系统动态矩阵, 存在一些核心测试集合 T , 使得不存在 $t \in T$ 长度大于 n ”。类似的陈述适用于核心历史 (Thm.2.5)。

其他框架模型和 PSR 之间的转换。虽然我们着重于通过系统动态向量和矩阵定义 PSR, 但也可以从其他模型推导出 PSR 模型。[Littman et al, 2002] 给出了将 POMDP 转换为等效 PSR 的算法, 并且 [James, 2005] 提出了将可解释的可观察算子模型 (OOM) 转换为 PSR 的附加算法。有趣的是, 没有从 PSR 恢复 POMDP 的已知方法 (可能因为存在多个 POMDP 映射到相同 PSR)。如果存在, 则用于 PSR 的任何学习算法将变成用于 POMDP 的学习算法。

13.3 PSR 模型学习

已经提出了许多算法来学习 PSR, 它们都解决了两个关键问题: 发现问题和学习问题 [James and Singh, 2004]。发现问题定义为寻找一组核心测试, 本质上是发现状态表示。学习问题定义为确定更新状态所需的模型参数, 本质上是学习系统动态方面的问题。在线性 PSR 的情况下, 这是所有一步测试和一步扩展的权重向量 m_{qi} 。

13.3.1 发现问题

可以通过研究其线性充分性确定核心测试: 一组核心测试对应于系统动态矩阵的一组线性无关的列, 因此来自线性代数的技术可以用于部分系统动态矩阵的经验估计。找到搜索线性无关列的算法是具有挑战性的任务, 因为矩阵的列是从数据中估计的, 而噪声列通常是

线性独立的 [Jaeger, 2004]。因此, 必须使用基于矩阵的奇异值的统计测试来估计矩阵的数值秩。整个过程通常依赖于矩阵的重复奇异值分解, 但这是没有效率的。例如, [James and Singh, 2004] 学习了一个“历史测试矩阵”, 它是系统动态矩阵的前身。它们的算法在达到停止标准前重复地估计矩阵的许多值。一种替代策略是避免完全搜索最低限度的充分数据集, 并通过使用大量随机采样测试来使用过度的基础信息 [Wingate and Singh, 2008], 但这种方法没有理论性质。 [428]

13.3.2 学习问题

发现核心测试后, 就可以学习更新参数。[Singh et al, 2003] 提出了学习问题的第一个算法, 其假设给出核心测试, 再使用梯度算法来计算权重。更常见的方法是使用回归和样本统计 [James and Singh, 2004]: 一旦给出一组核心测试, 更新参数可以通过回归出系统动态矩阵的适当项来求解。

一些作者将这两个问题组合成单个算法。例如, [Wiewiora, 2005] 提出了一种使用迭代扩展和回归方法学习正则形式 PSR 的方法, 而 [McCracken and Bowling 2006] 提出了一种基于梯度下降的在线发现和学习算法。

不是每一组矩阵 $M_{\alpha\alpha}$ 都能定义有效的线性 PSR。如果估计算法返回无效参数, 则 PSR 可能遭受下溢 (预测负概率) 或溢出 (预测大于 1 的概率)。[Wolfe, 2010] 确定了对 PSR 参数的精确约束, 其使用这些约束来改进参数的估计, 并且证明了所得模型产生的预测不会遭受下溢或溢出。

13.3.3 估计系统动态矩阵

许多学习和发现算法涉及估计系统动态矩阵, 并且通常使用样本统计。在具有复位的系统中, 学习器可以主动地重置为空历史, 以便重复地对样本项进行采样 [James and Singh, 2004]。在没有复位的系统中, 大多数研究人员使用后缀历史算法 [Wolfe et al, 2005] 来生成样本: 给定系统的轨迹, 我们将轨迹分割成所有可能的历史和未来。[Bowling et al, 2006] 提出的主动探索算法也适用于无复位的系统。

13.4 规划与 PSR

对 PSR 的规划的研究还较少, 部分是因为 PSR 研究者通常认为任何 POMDP 规划算法可以直接应用于 PSR (尽管没有正式的证据)。这部分是一些基础证明的结果, 其证明 PSR 和 POMDP 的价值函数具有类似的性质, 并且都与一般取样有关: “对于任何动态系统, POMDP 至 PSR 映射在恒定因素下保持邻域空间中的点之间的成对距离” [Izadi and Precup, 2008]。这是 PSR 与其在 13.2.9 节中讨论的等效 POMDP 之间的线性关系的结果, 意味着研究人员在将 POMDP 算法转换为 PSR 时没有什么困难。 [429]

[James et al, 2004] 建立了第一个基本的结论。他们表明可以为 PSR 构建策略树, 像 POMDP 一样, 并且 PSR 的价值函数是相对于预测向量的分段线性凸函数, 正如 POMDP 中的价值函数是相对于可信状态向量的分段线性凸函数一样。这推动了增量修剪和 Q 学习的 PSR 版本, 并同 PSR 其他版本一样与它们的 POMDP 相似。

[James et al, 2006] 证明了类似于 PERSEUS 的基于点的规划方法可以直接应用于 PSR; 再次, POMDP 和 PSR 性能相似。[Izadi and Precup, 2008] 获得了类似的结果, 他们提出了

另一个基于点的值迭代的 PSR 版本, 声称当 PSR 表示与相应的 POMDP 相比, 引入一定程度的压缩时, 他们的方法优于原始 PBVI, 并且, 结果是显著的。其他的研究表明, 使用记忆 PSR 的规划可以进一步提高性能并超过 vanilla PSR [James et al, 2004, 2006]。

策略梯度也已应用于 PSR, 结果不一。[Aberdeen et al, 2007] 在几个基准 PSR / POMDP 域中将自然演员—评论家 [Peters et al, 2005] 的策略梯度算法应用于 PSR, 结论是在两个模型下 PSR 似乎同样工作良好。[Wingate, 2008] 还将 NAC (以及最小二乘策略迭代 [Lagoudakis and Parr, 2003]) 应用于指数族 PSR (EFPSR), 结论是对于特定问题, 预测状态表示足够准确支持 NAC 但不能准确支持最小二乘策略迭代。[Boularias and Chaib-draa, 2009] 给出了部分可观察域和 PSR 中一个不寻常的规划组合, 他们观察到正如 PSR 可以取代 POMDP 作为动态模型, PSR 可以取代有限状态控制器 (FSC) 作为策略模型。他们将策略梯度应用于 PSR-FSC, 并证明它通常优于传统的 FSC。

总而言之, PSR 中的规划与 POMDP 中的规划无法确切地比较难易。也没有任何出色试验证明, 但一般来说, 使用 PSR 实现的规划比使用 POMDP 的规划更具有竞争力。虽然在利用 PSR 独特表示的领域内没有开发出特别新颖的方法, 但是许多作者已经得出结论, 利用 PSR 表示的规划将通常稍微好于或等于等同的 POMDP 表示的。

430

13.5 PSR 的扩展

基本的 PSR 模型已经以多种方式扩展, 主要是尽量将它们应用于更大的领域。这通常利用域中的某种结构来实现, 但是也可以利用更强大 (即非线性) 的状态更新方法来实现。

记忆 PSR。[James et al, 2005b] 表明, 记忆和预测表示的组合能获得比仅有预测表示更小的模型。他们的记忆 PSR (mPSR) “记住” 取样 (通常是最近的一个), 并且将不同记忆推导的未来取样和动作分布建立成单独的 PSR。如果记忆形成分割域的边界, 则变为紧凑的因子分解模型。[James and Singh, 2005a] 表明这个因子分解模型存在有效的规划算法。

分层 PSR。[Wolfe, 2009] 研究了分层 PSR 的时间抽象: 不是交织动作和取样, 而是交织选项 [Precup et al, 1998] 和观察。这些选项通过压缩动作—取样序列中的规则来捕获域中的时间结构, 这将使得较大的域能被缩放。例如, 在迷宫域中, 长走廊或大房间可以具有简单的压缩表示。Wolfe 分析了所产生的“选项级”动态系统作为 (对某些行和列进行求和) 系统动态矩阵的压缩。不过, 所得到的矩阵可以具有比原始系统更大的秩 (他提供了选项系统的秩不超过原始系统的秩的条件)。

因子 PSR。分层 PSR 利用域中的时间结构, 而因子 PSR [Wolfe, 2009] 利用的结构是一个在取样维度子集之间、通过利用条件独立性的、用向量值取样的动态系统的结构。这对于具有许多独立取样的域而言可能是重要的 (一个重要的示例是交通预测问题, 其中取样由高速公路上的数百辆汽车的信息组成)。因子 PSR 的一个重要特征是, 即使在对系统的未来状态进行长期预测时, 该表示仍可保持因子。这与 DBN 相反, 其中对许多时间步长后的潜在状态或取样的预测通常取决于当前所有潜在状态变量。Wolfe 解释了背后的关系:

“在当前时间步长考虑第 i 个潜在状态变量 (对于 DBN)。它将在下一个时间步长影响一些潜在状态变量的子集; 该子集将在随后的时间步长影响变量的另一个子集; 以此类推。一般来说, 第 i 个潜在状态变量的变量子集在不同时刻下将随着时间的前进而继续增长。因此, 一般来说, 不考虑 DBN 的置信状态 (跟踪潜在状态变量的联合分布), 并且潜在状态变量个数是指数级的。相反, 因子 PSR 不考虑潜在状态, 而只考虑动作和观察。”

431

多模式 PSR。多模式 PSR (MMPSR) 模型 [Wolfe et al, 2008, 2010] 应用在几种操作模式之间切换的不受控制的系统中。多模式 PSR 以当前模式为条件进行专门的预测, 这可以简化整体模型。MMPSR 的灵感来自预测汽车在高速公路上的移动问题, 其中汽车通常以离散操作模式操作, 例如改变车道或者超车。这些模式类似于选项, 但是模式不是潜在的、不可观测的量 (相对于例如层级 HMM 中的上层)。Wolfe 定义了可识别性条件来识别模式, 这可以是过去和未来取样的函数, 但这并不限制可以建模的系统的类别, 它仅影响有利于系统建模的这一组模式。

关系 PSR。[Wingate et al, 2007] 展示了如何通过将所有谓词在未来预测中接地来使用 PSR, 并在关系域中建模。这项工作的一个贡献是引入了新的测试: 集合测试 (其对动作序列和取样集合进行预测) 和索引测试 (其对动作和取样序列预测, 其中在时刻 t 的动作 / 取样需要具有与时间 $t + k$ 时的动作 / 取样值相同但未指定的值, 因此在测试中允许存在“变量”)。这些测试有很高的灵活性, 但仍然保留线性 PSR 的所有线性特性, 使其具有表达性和易处理性。

连续 PSR。对连续动作和取样的系统研究较少。[Wingate and Singh, 2007b] 提出了对 PSR 的泛化, 其中测试预测了一系列连续动作和取样的密度。这产生了一个问题: 具有离散取样的 PSR 通过系统动态矩阵的秩导出充分数据集的概念从而确定状态, 但是对于连续取样和动作的系统, 这样的矩阵及其秩不再存在。作者为连续情况定义了一个类似的结构, 称为系统动态分布, 并使用信息理论概念来定义一个充分数据集, 从而确定状态。它们使用核密度估计来学习模型, 并使用信息梯度来直接优化状态表示。

其他扩展。[Tanner et al, 2007] 提出了一种从低级状态表示学习高级抽象特征的方法。[Rudary and Singh, 2004] 表明, 在状态更新侧允许存在非线性充分数据集时, 线性模型可以被压缩。

13.6 其他具有预测性定义状态的模型

还有其他动态系统模型, 通过使用对未来的预测而确定状态。

432

13.6.1 可观测算子模型

可观测算子模型 (OOM) 由 [Jaeger, 2000] 引入和研究。像 PSR 一样, 在同一个基本主题上有几个变体, 使它更像是一个框架而不是单个模型。OOM 类的模型可用于处理不同版本的动态系统: 基础 OOM 模型应用于不可控动态系统, 而 IO-OOM 模型应用于可控动态系统。OOM 与 PSR 有几种相似之处。例如, 存在核心测试 (“特征事件”)、核心历史 (“指示性事件”) 以及与系统动态矩阵类似的结构。OOM 将未来的预测向量表示为状态, 但预测不只是对应于单个测试。相反, 状态向量中的每个元素是对等长 k 的测试集的预测。这些测试集有约束: 它们必须互不相交, 且它们的并集必须覆盖长度 k 的所有测试。

对 IO-OOM 的显著限制是测试中使用的动作序列必须对于所有测试是相同的, 这是满足关于状态向量的一些假设所需要的。这个假设足够严格 [James (2005)], 这导致对于 IO-OOM 不能建模的系统, PSR 可以建模。还存在不使用预测作为其状态表示的一部分的 OOM 的变体 (称为 “不可解释 OOM”), 但是没有用于这些模型的学习算法 [James, 2005]。我们请读者参见 [Jaeger, 2004] 的技术报告, 以详细比较 PSR 和 OOM。

13.6.2 预测线性高斯模型

“预测性线性高斯”模型 (PLG) [Rudary et al, 2005] 是具有线性动态和简单标量取样的不可控系统对 PSR 的泛化模型。PLG 将状态定义为在未来短期内的窗口上的高斯分布参数。这是 PSR 表示的另一个泛化：状态定义为未来动作和取样的均值和协方差矩阵，两者都可以解释为对未来特征的期望。

PLG 是预测性定义模型的另一个例子，其代表性地等同于其经典模型。PLG 在建模能力上与线性动态系统在形式上是等价的：在任何时间 t ，线性动态系统或 PLG 可以用于预测未来观察的分布，并且这些分布是相同的。此外，PLG 与相应的线性动态系统一样紧凑：可以通过 n 维 PLG 捕获 n 维线性动态系统，并且可以使用与标准卡尔曼滤波器更新方程 [Kalman, 1960] 相等计算复杂度的方程来维持 PLG 中的状态。基本 PLG 已经以许多方式扩展，包括向量值观测 [Rudary and Singh, 2008] 和线性控制 [Rudary and Singh, 2006]。[433] PLG 对于线性二次调节器的最优策略是状态的线性函数（正如经典 LDS），并且其可以利用与 Riccati 方程类似的方程找到。与 PSR 类似，PLG 具有基于样本统计和回归的学习算法。[Rudary et al, 2005] 获得了一个重要的学习能力结论，PLG 与 PSR 拥有一致的参数估计算法，这对基于 EM [Ghahramani and Hinton, 1996] 等方法的典型线性动态系统学习非常重要。

非线性动态可以有两种不同的方式表示。核心 PLG [Wingate and Singh, 2006a] 模型通过使用核心技巧，可以将线性动态表示在非线性特征空间。更受欢迎的和可学习的方法是假设动态是分段线性的，即“混合 PLG”模型 [Wingate and Singh, 2006b]。在许多方面，这些扩展与卡尔曼滤波器的开发并行：例如，可以使用基于 σ 点近似的有效近似推理算法来更新 KPLG 中的状态（产生与无迹卡尔曼滤波器相关的算法 [Wan and van der Merwe, 2000]）。

13.6.3 时序差分网络

[Sutton and Tanner, 2005] 的时序差分网络模型是 PSR 的重要泛化。在 TD 网络中，如 PSR 一样，状态表示为关于未来的一组预测。然而，明确地允许这些预测以组合递归方式彼此依赖。这表明时序差分算法与 PSR 使用的蒙特卡罗方法相反，其可用于学习预测，正是这些算法形成了模型的基础。在学习过程中，测试的递归性质和时序差分方法的使用自然地概括为包括引入资格迹来形成 TD (λ) 网络的学习的多步备份 [Tanner and Sutton, 2005a]。

虽然 TD 网络在理论上是有吸引力的，但它们没有 PSR 等效的严谨的分析力。它们的表示能力及它们的状态更新机制的都有所欠缺。例如，TD 网络使用与单层神经网络相关的一般非线性状态更新机制，尽管这不是模型的基本组成部分。它也可能使用其他状态更新，但不清楚其是否与贝叶斯定律规定的统计上最优的更新相关。相反，PSR 明确地使用贝叶斯定律规定的最优状态更新机制。

经验上，TD 网络与 PSR 有相同的成功率和失败率，该模型适用于相当小的域中。虽然在一般的 TD 网络上的研究较少，但是 TD 网络和 PSR 的学习算法的开发在某些方面彼此并行。例如，[Tanner and Sutton, 2005b] 提出在状态表示中包括一些历史，以 [James and Singh, 2005a] 提出的记忆 PSR 的方式学习，提高学习效果。[434]

13.6.4 分集自动机

[Rivest and Schapire, 1987] 的分集自动机是基于未来预测的模型，它有一些严格的限制。与 PSR 模型一样，分集模型将对未来的预测向量表示为状态。然而，这些预测不像

PSR 通常的测试那样灵活, 而是限于像 [Rudary and Singh, 2004] 使用的 e 测试一样。每个测试 t_i 是给定一串动作 n_i 但在时间 $t+1$ 和 $t+n_i$ 之间没有给出任何取样值的情况下, 在 n_i 步骤中将发生某个取样的概率。这些测试中的每一个对应于关于未来取样分布的等价类。

[Rivest and Schapire, 1987] 提出分集模型所需的测试数量相对于最小 POMDP 需要的状态数量存在的紧密界限。分集模型可以压缩或膨胀系统: 在最好的情况下, 需要对数数量的测试, 但在最坏的情况下, 需要指数数量的测试。这与 PSR 形成对比, 其中对由 n 个状态 POMDP 建模的任何域进行建模只需要 n 个测试。分集模型也限于具有确定性转换和确定性取样的系统。这是由于模型的状态更新机制以及通过将模型限制为有限数量的未来分布的等价类来将模型限制为有限数量的测试。

13.6.5 指数族 PSR

指数族 PSR (EFPSR) 是一个将许多其他模型与预测性定义状态统一的通用模型 [Wingate and Singh, 2007a]。有观察结果表明, 这些模型跟踪未来短期内的充分数据集是以指数分布的。例如, PLG 使用高斯参数, 而 PSR 使用多项式参数。指数族分布是可以被写为 $p(\text{future}|\text{state}, \text{past}) \propto \exp\{w_i^T \phi(\text{future})\}$ 的任意分布; $\phi(\text{future})$ 以及 w_i 更新机制的不同导致不同的模型。

于是可以提出将概率指数族分布置于未来短期内观测中, 并将未来的特征 ϕ 参数化。这种概括已经用于预测和分析新模型, 包括设计用于对具有大量特征的域进行建模的线性—线性 EFPSR [Wingate and Singh, 2008], 并且还产生了 PLG 的信息形式 [Wingate, 2008]。并提出图形模型、最大熵建模和 PSR 之间的强连接。

纯粹的说, EFPSR 并不真的是一个 PSR: 虽然表示是根据未来的短期分布来定义的, 但是模型的参数不太直接地依赖于数据: 它们不能用与未来事件的期望值相同的方式进行验证。因此, 该模型位于潜在状态空间模型和 PSR 之间。

13.6.6 转换 PSR

435

转换 PSR [Rosencrantz et al, 2004] 通过估计系统动态矩阵来学习模型。然而, 它们不搜索线性无关的列 (即核心测试), 而是对系统动态矩阵执行奇异值分解 (可以解释为核心测试的线性组合) 以恢复状态的低维表示。其得到状态是 PSR 的旋转结果, 但是状态向量的元素不能解释为测试的预测。[Boots et al, 2010] 随后采用替代学习算法和处理连续观察的能力对这项工作进行了泛化。结果是得到在统计上一致的第一个学习算法, 以及哪些经验工作足以支持复杂域中的规划。

13.7 总结

虽然有了一些理论和经验结果, 但预测性定义状态表示模型仍然相对年轻。我们可以从这些结果中得出什么广泛的结论? 该模型未来可能有什么?

关于紧凑性、表达性和可学习性的研究结果都表明 PSR 模型是可行的。例如, 我们已经看到在 PSR 和 POMDP 之间存在强连接: PSR 至少与 POMDP 一样紧凑, 并且具有可观的计算复杂度, 以及更具表现力。与其他模型一样, PSR 可以利用域中的结构, 如通过因子分解、层次化、交换、内核化或连续域的扩展论证。当然, 这个模型真正的突出点是学习能力, 我们已经看到了各种功能性学习算法, 以及在 PLG 和 TPSR 的统计一致性结果中的一

些令人瞩目的最佳学习能力。

预测表示模型也出乎意料地灵活。不同的作者通过使用特定测试的概率 (PSR 模型)、特定测试的密度 (连续 PSR 模型)、未来短期内特征的预期 (PLG 族模型), 未来短期内观测分布的自然参数 (EFPSR), 以及各种测试 (包括集合测试、索引测试和选项测试) 来捕获状态。似乎还有很多其他尚未开发的可能性。

同样重要的就是未出现严重的负面结果。随着研究人员探索这个模型, 总是有一个问题: 有什么是预测性定义状态表示模型不能做的吗? 这种表示会有一个限制吗? 到目前为止, 答案是否定的: 对预测性定义状态表示没有已知的代表性、计算性、理论性或经验性的限制, 事实上, 仍然有很好的理由去相信预测性定义状态表示对不同的任务具有独特的属性。

436

预测性定义状态表示的深度已经全部探测出了吗? 似乎不大可能。但也许它们最重要的贡献之一就是: 它们激励着我们重新思考状态的学习及表示到底意味着什么。

参考文献

- Aberdeen, D., Buffet, O., Thomas, O.: Policy-gradients for psrs and pomdps. In: International Workshop on Artificial Intelligence and Statistics, AISTAT (2007)
- Astrom, K.J.: Optimal control of Markov decision processes with the incomplete state estimation. *Journal of Computer and System Sciences* 10, 174–205 (1965)
- Boots, B., Siddiqi, S., Gordon, G.: Closing the learning-planning loop with predictive state representations. In: *Proceedings of Robotics: Science and Systems VI*, RSS (2010)
- Boularias, A., Chaib-draa, B.: Predictive representations for policy gradient in pomdps. In: *International Conference on Machine Learning, ICML* (2009)
- Bowling, M., McCracken, P., James, M., Neufeld, J., Wilkinson, D.: Learning predictive state representations using non-blind policies. In: *International Conference on Machine Learning (ICML)*, pp. 129–136 (2006)
- Ghahramani, Z., Hinton, G.E.: Parameter estimation for linear dynamical systems. Tech. Rep. CRG-TR-96-2, Dept. of Computer Science, U. of Toronto (1996)
- Izadi, M., Precup, D.: Model minimization by linear psr. In: *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1749–1750 (2005)
- Izadi, M.T., Precup, D.: Point-Based Planning for Predictive State Representations. In: Bergler, S. (ed.) *Canadian AI. LNCS (LNAI)*, vol. 5032, pp. 126–137. Springer, Heidelberg (2008)
- Jaeger, H.: Observable operator processes and conditioned continuation representations. *Neural Computation* 12(6), 1371–1398 (2000)
- Jaeger, H.: Discrete-time, discrete-valued observable operator models: A tutorial. Tech. rep., International University Bremen (2004)
- James, M., Singh, S., Littman, M.: Planning with predictive state representations. In: *International Conference on Machine Learning and Applications (ICMLA)*, pp. 304–311 (2004)
- James, M., Wessling, T., Vlassis, N.: Improving approximate value iteration using memories and predictive state representations. In: *Proceedings of AAAI* (2006)
- James, M.R.: Using predictions for planning and modeling in stochastic environments. PhD thesis, University of Michigan (2005)
- James, M.R., Singh, S.: Learning and discovery of predictive state representations in dynamical systems with reset. In: *International Conference on Machine Learning (ICML)*, pp. 417–424 (2004)
- James, M.R., Singh, S.: Planning in models that combine memory with predictive representations of state. In: *National Conference on Artificial Intelligence (AAAI)*, pp. 987–992 (2005a)
- James, M.R., Wolfe, B., Singh, S.: Combining memory and landmarks with predictive state



- representations. In: International Joint Conference on Artificial Intelligence (IJCAI), pp. 734–739 (2005b)
- Kalman, R.E.: A new approach to linear filtering and prediction problem. *Transactions of the ASME—Journal of Basic Engineering* 82(Series D), 35–45 (1960)
- Lagoudakis, M.G., Parr, R.: Least-squares policy iteration. *Journal of Machine Learning Research (JMLR)* 4, 1107–1149 (2003)
- Littman, M.L., Sutton, R.S., Singh, S.: Predictive representations of state. In: *Neural Information Processing Systems (NIPS)*, pp. 1555–1561 (2002)
- McCracken, P., Bowling, M.: Online discovery and learning of predictive state representations. In: *Neural Information Processings Systems (NIPS)*, pp. 875–882 (2006)
- Nikovski, D.: State-aggregation algorithms for learning probabilistic models for robot control. PhD thesis, Carnegie Mellon University (2002)
- Peters, J., Vijayakumar, S., Schaal, S.: Natural actor-critic. In: *European Conference on Machine Learning (ECML)*, pp. 280–291 (2005)
- Precup, D., Sutton, R.S., Singh, S.: Theoretical results on reinforcement learning with temporally abstract options. In: *European Conference on Machine Learning (ECML)*, pp. 382–393 (1998)
- Rafols, E.J., Ring, M.B., Sutton, R.S., Tanner, B.: Using predictive representations to improve generalization in reinforcement learning. In: *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 835–840 (2005)
- Rivest, R.L., Schapire, R.E.: Diversity-based inference of finite automata. In: *IEEE Symposium on the Foundations of Computer Science*, pp. 78–87 (1987)
- Rosencrantz, M., Gordon, G., Thrun, S.: Learning low dimensional predictive representations. In: *International Conference on Machine Learning (ICML)*, pp. 695–702 (2004)
- Rudary, M., Singh, S.: Predictive linear-Gaussian models of stochastic dynamical systems with vector-value actions and observations. In: *Proceedings of the Tenth International Symposium on Artificial Intelligence and Mathematics, ISAIM* (2008)
- Rudary, M.R., Singh, S.: A nonlinear predictive state representation. *Neural Information Processing Systems (NIPS)*, 855–862 (2004)
- Rudary, M.R., Singh, S.: Predictive linear-Gaussian models of controlled stochastic dynamical systems. In: *International Conference on Machine Learning (ICML)*, pp. 777–784 (2006)
- Rudary, M.R., Singh, S., Wingate, D.: Predictive linear-Gaussian models of stochastic dynamical systems. In: *Uncertainty in Artificial Intelligence*, pp. 501–508 (2005)
- Shatkay, H., Kaelbling, L.P.: Learning geometrically-constrained hidden Markov models for robot navigation: Bridging the geometrical-topological gap. *Journal of AI Research (JAIR)*, 167–207 (2002)
- Singh, S., Littman, M., Jong, N., Pardoe, D., Stone, P.: Learning predictive state representations. In: *International Conference on Machine Learning (ICML)*, pp. 712–719 (2003)
- Singh, S., James, M.R., Rudary, M.R.: Predictive state representations: A new theory for modeling dynamical systems. In: *Uncertainty in Artificial Intelligence (UAI)*, pp. 512–519 (2004)
- Sutton, R.S., Tanner, B.: Temporal-difference networks. In: *Neural Information Processing Systems (NIPS)*, pp. 1377–1384 (2005)
- Tanner, B., Sutton, R.: Td(λ) networks: Temporal difference networks with eligibility traces. In: *International Conference on Machine Learning (ICML)*, pp. 888–895 (2005a)
- Tanner, B., Sutton, R.: b) Temporal difference networks with history. In: *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 865–870 (2005)
- Tanner, B., Bulitko, V., Koop, A., Paduraru, C.: Grounding abstractions in predictive state representations. In: *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1077–1082 (2007)
- Wan, E.A., van der Merwe, R.: The unscented Kalman filter for nonlinear estimation. In: *Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control* (2000)



- Wiewiora, E.: Learning predictive representations from a history. In: International Conference on Machine Learning (ICML), pp. 964–971 (2005)
- Wingate, D.: Exponential family predictive representations of state. PhD thesis, University of Michigan (2008)
- Wingate, D., Singh, S.: Kernel predictive linear Gaussian models for nonlinear stochastic dynamical systems. In: International Conference on Machine Learning (ICML), pp. 1017–1024 (2006a)
- Wingate, D., Singh, S.: Mixtures of predictive linear Gaussian models for nonlinear stochastic dynamical systems. In: National Conference on Artificial Intelligence (AAAI) (2006b)
- Wingate, D., Singh, S.: Exponential family predictive representations of state. In: Neural Information Processing Systems, NIPS (2007a) (to appear)
- Wingate, D., Singh, S.: On discovery and learning of models with predictive representations of state for agents with continuous actions and observations. In: International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1128–1135 (2007b)
- Wingate, D., Singh, S.: Efficiently learning linear-linear exponential family predictive representations of state. In: International Conference on Machine Learning, ICML (2008)
- Wingate, D., Soni, V., Wolfe, B., Singh, S.: Relational knowledge with predictive representations of state. In: International Joint Conference on Artificial Intelligence (IJCAI), pp. 2035–2040 (2007)
- Wolfe, B.: Modeling dynamical systems with structured predictive state representations. PhD thesis, University of Michigan (2009)
- Wolfe, B.: Valid parameters for predictive state representations. In: Eleventh International Symposium on Artificial Intelligence and Mathematics (ISAIM) (2010)
- Wolfe, B., James, M.R., Singh, S.: Learning predictive state representations in dynamical systems without reset. In: International Conference on Machine Learning, pp. 980–987 (2005)
- Wolfe, B., James, M., Singh, S.: Approximate predictive state representations. In: Proceedings of the 2008 International Conference on Autonomous Agents and Multiagent Systems (AAMAS) (2008)
- Wolfe, B., James, M., Singh, S.: Modeling multiple-mode systems with predictive state representations. In: Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems (2010)



博弈论和多学习器强化学习

Ann Nowé, Peter Vrancx, Yann-Michaël De Hauwere

摘要

强化学习最初是为马尔可夫决策过程开发的。它允许单个学习器在随机静态环境中学习可能的延迟回报信号最大化的策略。如果学习器能进行充足的实验且其实验环境是马尔可夫的，强化学习能保证最优策略的收敛性。然而，多学习器在共享环境中应用强化学习时，可能会超出马尔可夫决策过程模型的范围。在这样的系统中，学习器的最优策略不仅仅取决于环境，还取决于其他学习器采取的策略。这些情况在其他领域也会出现，例如，机器人学、通信行业、经济学、分布控制、拍卖、交通信号灯控制等。多学习器学习应用于这些领域，或由于其领域的复杂性，或因为其控制机制本质上是非中心化。在这样的系统中，学习器通过与其他学习器合作或者让它们竞争来发现这个问题的好的解决办法是非常重要的。本章主要讲强化学习技术在多学习器系统中的应用。我们描述了一个基于博弈论的经济研究基础学习结构，并阐述了这个系统额外的复杂性。我们还为不同领域的多学习器强化学习研究描述了一个典型的选择算法。

14.1 简介

强化学习技术的学习贯穿了整个章节，它能够通过实验和错误的交互以及它的环境使单个学习器学习最优行为。各种 RL 技术已先后被开发出来了，使学习器在各种情况下优化其动作。然而，当多学习器在一个共享的环境中同时应用强化学习时，传统的解决方案就不适用了。在多学习器的设定中，经常违反保证收敛所需的假设。尽管在大多数基础情况中学习器会共享稳定的环境并为了单一状态需求学习一个策略，但仍会导致出现许多新的复杂性问题。当学习器的目标一致且所有的学习器都尝试最大化相同的回报信号时，协调会被一直请求直到达到全局最优。当学习器有对立的目标时，可能不存在明确的最佳解决方案。在这种情况下，在学习器策略之间的平衡被频繁检索。在这样的平衡中，当其他的学习器保持它们静态的动作时所有学习器都不能改进它的回报。

[441]

除了多学习器之外，当我们假设一个请求多个连续策略的动态环境时，这个问题就变得更加复杂。现在的学习器做的不仅仅是协调，它们还必须考虑到当前状态的环境。这个问题变得越来越复杂化，因为学习器事实上只是有限的系统信息。在通常情况下，虽然这些动作与它们拥有的回报和它们的环境有直接的影响，但它们没有能力去观察来自其他学习器的动作回报。在最极端的情况中，学习器可能并不知道其他学习器的存在，所以会使环境看起来不稳定。在其他的情况中，学习器可以访问所有的信息，这都是由于计算复杂性和学习器之间所需的协调。为了得出一个成功的多学习器方法，需要解决所有上述问题。图 14.1 所描述的是多学习器强化学习的标准模型。



尽管增加了学习复杂度，但是仍然存在对多学习器系统的需求。通常系统本质上是离散的并且中央的单一学习器的学习方法是不可行的。可能出现这种情况是因为会遇到多重的、可能冲突的目标，或者数据或控制的物理分布，或者仅仅是因为单个集中控制器需要许多资源。这种系统的案例有多机器人、分布式网络路由、分布式负载均衡、电子拍卖、交通控制等。

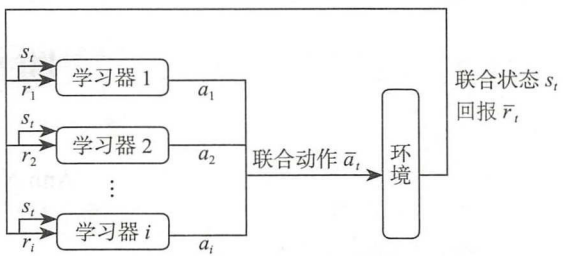


图 14.1 多学习器在相同环境中的演示

为了适应多学习器系统的需求，组成相互作用学习器的复杂性，推动多学习器强化学习领域的发展，它由两根基柱构成：人工智能中强化学习的研究和跨学科工作中的博弈论。然而早前的博弈论纯粹是注重于竞技性游戏，它之后就发展成一个为了分析策略交互的常用框架。它已经吸引了各个领域比如心理学、经济学和生物学的兴趣。随着多学习器系统的出现，它也在人工智能团体和计算机科学之中获得了重要意义。本章谈论博弈论是怎样提供一个为多学习器学习阐述问题设定的方法和分析学习结果的工具。

本章考虑的多学习器系统的特点是学习器之间策略的相互作用。我们的意思是，这些学习器是独立存在的实体，它们有各自的目标和独立的决策能力，但也受到彼此策略的影响。我们将此设定通过可视为分布式或并行加强学习的方法进行区分。在相同系统中的多重学习器就可以共同地学习简单的目标。这包括系统的多重学习器策略同步更新 [Mariano and Morales,2001]、群基础技术 [Dorigo and Stützle, 2004] 和处理方法区分的学习器中的学习状态空间 [Steenhaut et al, 1997]。像 [Tsitsiklis, 1994] 框架所描述的一样，这样的系统中许多可以视为标准强化学习的高级探索技术仍覆盖于单一学习器理论框架。只要最后弃用过时的信息，这个算法的收敛就仍是有效的。比如，它能在标准 Q 学习更新规则（见第 1 章）右侧的最大算子使用过时的 Q 值。有意思的是当它的 Q 值属于不同的学习器时学习器都会探索环境中它们自己的每个部分，然后交换它们的 Q 值。在本章中涵括的系统超出标准的单一学习器理论，因此需要一个不同的框架。

表 14.1 给出了一个基于策略交互的多学习器的概念研究。列出的技术是根据它们在多学习器系统中学习时的适用性和信息种类进行分类的。我们对无状态博弈的技术做了区分，这个技术着重的是处理当假设环境是稳定时的多学习器交互和马尔可夫博弈技术（用于处理多学习器和动态环境的交互）。此外，我们也通过学习器的学习来展示信息的使用。独立的学习器学习仅仅基于它们拥有的回报观察，而联合动作学习器还会使用其他学习器的动作观察和可能的回报。

表 14.1 常用 MARL 方法的概述。算法是按照它们的适应性（共同特点或者马尔可夫博弈）和它们的信息需求（标量反馈或者接合动作信息）进行分类

		博弈环境		
		无状态博弈	团队马尔可夫博弈	一般的马尔可夫博弈
信息需求	独立的学习器	无状态 Q 学习 学习自动机 IGA FMQ 委托序列	策略迭代 策略梯度	MG-ILA (WoLF)-PG 学习协调 独立的 RL CQ 学习



(续)

		博弈环境		
		无状态博弈	团队马尔可夫博弈	一般的马尔可夫博弈
信息需求	联合动作学习器		分布式 Q 学习 稀疏表 Q 学习 有效的协调	纳什 Q 学习 是敌是友 Q 学习 联合动作学习 相关联的 Q 学习

在下文中我们会阐述重复博弈框架。这个设定介绍许多由学习学习器之间的相互作用产生的复杂性。然而，重复博弈设置仅考虑在静态的无状态环境，其中学习挑战仅来自与其他学习器的交互。在 14.3 节中我们会介绍马尔可夫博弈，该框架概括了通常用于单一学习器 RL 的马尔可夫决策过程 (MDP) 的设定。为了求解这些马尔可夫博弈，我们会讲解值迭代和策略迭代的方法。14.4 节描述的是现在最新的多学习器的研究，研究会介绍独立学习技术和在完全联合状态的联合动作空间中马尔可夫博弈技术操作的核心部分。最后在 14.5 节，我们会简短地描述其他有意思的背景材料。

444

14.2 重复博弈

14.2.1 博弈论

博弈论的中心思想是为一组玩家之间的博弈建立一个策略交互模型。一次博弈就相当于一个数学模型，阐述的是玩家在根据各自制定的奖励策略中的交互结果。一次博弈可能有不同的表现。比如，传统的 AI 研究往往侧重于扩展型博弈，它们可用来表示玩家轮流执动作的情况。这个现象应用在各个方面，比如，古典的极小极大算法 [Russell and Norvig, 2003]。然而，在本章中，我们将着重于所谓的标准型博弈，其中游戏玩家通过同时选择独立的动作来展示。这个设定经常用作一个多学习器学习方法的测试平台。下面我们回顾基本的博弈论术语并定义一些在博弈中的常见的解决方案。

14.2.1.1 标准型博弈

定义 14.1 一个标准型博弈是一个元组 $(n, A_1, \dots, A_n, R_1, \dots, R_n)$ ，其中

- $1, \dots, n$ 是一个在游戏中的参与者收集器，叫作玩家。
- A_k 是玩家 k 可用的一组（有限的）动作。
- $R_k: A_1 \times \dots \times A_n \rightarrow \mathbb{R}$ 是玩家 k 的个人回报函数，说明它从一场游戏 $\mathbf{a} \in A_1 \times \dots \times A_n$ 中应该获得的回报。

一场博弈通过每一个玩家 k 从自己的动作集合 A_k 去单独选择一个个人的动作 a 来演示。所有玩家动作的组合构成一个联合动作或学习器动作设定 $\mathbb{A} = A_1 \times \dots \times A_n$ 中的动作组合，来表示联合动作 $\mathbf{a} \in \mathbb{A}$ ， $R_k(\mathbf{a})$ 表示学习器 k 的预期支付。

标准型博弈是以它们的支付矩阵表示的。表 14.2 给出了一些典型的双人博弈。在这种情况下，玩家 1 指定矩阵中的一行，玩家 2 决定其列来选择动作。然后在矩阵里相应的条目给玩家 1 支付且玩家 2 回报。玩家 1 和 2 也分别叫作行和列玩家。使用更多的空间矩阵描述 n 玩家博弈，矩阵中的每个条目都包含对应于每个学习器的动作组合的支付。

策略 $\sigma_k: A_k \rightarrow [0, 1]$ 是 $\mu(A_k)$ 的一个元素，是在玩家 k 的动作集合 A_k 上的概率分布集合。如果一些动作 $\sigma_k(a) = 1 (a \in A_k)$ 且其他所有动作都为 0 则该策略叫作单纯策略，否则就叫作



445

混合策略。策略组合 $\sigma = (\sigma_1, \dots, \sigma_n)$ 是策略的向量，包涵了所有玩家的策略。如果所有在 σ 中的策略是单纯策略，这个策略组合就符合联合动作 $\mathbf{a} \in \mathbb{A}$ 。在标准化博弈中的重要设想是在玩家策略中的预期收益是线性的，也就是说玩家 k 的策略组合的预期回报是通过下式给出的：

$$R_k(\sigma) = \sum_{\mathbf{a} \in \mathbb{A}} \prod_{j=1}^n \sigma_j(a_j) R_k(\mathbf{a})$$

其中 a_j 是玩家 j 在组合 \mathbf{a} 中的动作。

14.2.1.2 博弈的类型

根据玩家的回报函数，可以进行博弈的分类，当所有的玩家共享相同的回报函数时，博弈称为相同支付或共同利益博弈。如果所有的玩家的回报加起来为 0 则博弈称为零和博弈。在后面的博弈中某一个玩家的胜利会把其他对立玩家转化成损失。因此这些博弈也称为纯竞技博弈。当考虑博弈无特殊限制时我们叫作一般和博弈。

表 14.2 以 2 个动作的双人博弈为例。从左到右看：a) 猜硬币，纯竞技（零和）博弈；b) 囚徒困境，一般和博弈；c) 协调游戏，共同利益（相同支付）博弈；d) 性别之战，学习器有不同选择的协调游戏（纯纳什均衡用粗体表示）

	a1	a2		a1	a2		a1	a2		a1	a2
a1	(1,-1)	(-1,1)	a1	(5,5)	(0,10)	a1	(5,5)	(0,0)	a1	(2,1)	(0,0)
a2	(-1,1)	(1,-1)	a2	(10,0)	(1,1)	a2	(0,0)	(10,10)	a2	(0,0)	(1,2)
a)			b)			c)			d)		

以表 14.2 中所说的这些博弈类型为例。在表中的第一个猜硬币的博弈是一个严格的竞技博弈，这个博弈描述在这情况中两个玩家必须要各自选出硬币的一面去展示（也就是正面或者反面）。当两个玩家都是一样的面，玩家 1 获胜玩家 2 要支付 1 个单位。当硬币是不一样的面，玩家 2 获胜并收到玩家 1 支付的 1 个单位。由于两个玩家博弈时互为对手，一个玩家的胜利自动转化为另一个玩家的损失，因此这叫作零和博弈。

在表 14.2 的第二个博弈叫作囚徒困境，是一个普通的一般和博弈。在这个博弈中，两个罪犯因为犯罪被逮捕。他们的动作有两种可能：一起合作否认罪行（动作 a1），或者背叛出卖另一个，指控是他在犯罪（动作 a2）。如果他们合作否认罪行，警察因为没有充分的证据只能给他们最轻的判决，变成两个人都有支付 5。如果一个人否认但一个人背叛，这个否认的人会获得所有罪行（支付 0），然而另一个背叛者则逃脱了惩罚（支付 10）。然而，如果两个人都背叛，那么他们都将接受审判（支付 1）。在这个博弈中的问题中合作动作是完全服从于背叛动作的：无论另一个人选择什么，背叛总会有高奖励。尽管事实上，两个玩家可以同时更好地发挥双方合作，但通常还是会导致（背叛，背叛）的结果。

446

表 14.2 中的第三个博弈是共同利益博弈。在这个例子中两个玩家在每一个联合动作中都获得了相同的支付。这个博弈的挑战性是玩家通过合作达到最佳联合动作。选择错误的联合动作得出次优的支付和失败的合作会导致 0 支付。

第四个博弈，性别之战，是另一个协调博弈的例子。然而在这里，玩家给出各自的回报并提出不同的结果。学习器 1 提出 (a1, a2)，然而学习器 2 提出 (a2, a2)。除了协作问题之外，玩家还要立即同意所提出的结果。

当然博弈不仅限于只有两个动作，它可以是任何数量的动作。在表 14.3 中我们会展示三个动作的共同利益博弈。首先，[Claus and Boutilier, 1998] 的攀岩游戏中，纳什均衡的矩



阵外面加了层惩罚因子。在第二个博弈中，因为参数 $k < 0$ 所以抛弃惩罚。 k 越小，通过对优选解决方案 ((a1, a1) 和 (a3, a3)) 的学习而达成一致变得更为困难 (在 [Claus and Boutilier, 1998] 中，动态博弈使用迭代值的方法进行分析，见 14.2.2 节)。

表 14.3 以双人、三动作博弈为例。从左到右：a) 攀岩游戏；b) 惩罚博弈，这里 $k \leq 0$ 。

两个博弈都是共同利益类型的。黑体部分表示纳什均衡

	a1	a2	a3		a1	a2	a3
a1	(11,11)	(-30,-30)	(0,0)	a1	(10,10)	(0,0)	(k,k)
a2	(-30,-30)	(7,7)	(6,6)	a2	(0,0)	(2,2)	(0,0)
a3	(0,0)	(0,0)	(5,5)	a3	(k,k)	(0,0)	(10,10)

a)

b)

14.2.1.3 博弈解决思路

由于博弈中的玩家具有取决于其他玩家的动作的个人回报函数，所以通常不能明确地给出确定的博弈期望结果。不能简单地期望参与者最大化他们的支付，因此不可能所有玩家同时达到这个目的。例如，参见表 14.2d 中的“性别之战”。

447

这种学习情境的一个重要概念是最佳回应。当做出一个最佳回应时，玩家可以最大程度地利用他对手在博弈中的当前策略的回报。也就是说，如果在博弈中的其他部分保持策略不变，玩家就不可能提高它的回报。从形式上看，我们对这个概念的定义如下。

定义 14.2 令 $\sigma = (\sigma_1, \dots, \sigma_n)$ 为策略情况，令 σ_{-k} 表示相同的策略概况，但没有玩家 k 的策略 σ_k 。如果下式成立，则称策略 $\sigma_k^* \in \mu(A_k)$ 为对于玩家 k 的最佳响应：

$$R_k(\sigma_{-k} \cup \sigma_k^*) \geq R_k(\sigma_{-k} \cup \sigma_k') \forall \sigma_k' \in \mu(A_k)$$

这里 $\sigma_{-k} \cup \sigma_k'$ 表示策略概况，除了演示 σ_k' 的学习器 k ，其中所有学习器在 σ 中执行相同的策略，也就是 $(\sigma_1, \dots, \sigma_{k-1}, \sigma_k', \sigma_{k+1}, \dots, \sigma_n)$ 。 k 博弈中解决方案的核心概念是纳什均衡 (NE)。在一个纳什均衡中，玩家们都会发挥最佳回应，意味着每一个玩家都要对当前其他玩家的策略做最佳回应。基于最佳回应的概念，我们定义纳什均衡如下。

定义 14.3 如果对于每一个玩家 k ，策略 σ_k 都是对其他玩家的策略的最佳响应 σ_{-k} ，则策略集合 $\sigma = (\sigma_1, \dots, \sigma_n)$ 称为纳什均衡。

因此，当处理一个纳什均衡时，没有玩家可以通过单方面偏离均衡策略配置来改进回报。因此没有玩家会改变其策略的动机，因为多个玩家必须同时改变他们的策略，才可脱离纳什均衡。

如表 14.2c 所示，在共同利益博弈中，纳什均衡为所有玩家调整出一个局部最优，但是它不一定符合全局最优。这在协调博弈中可以清楚地看出，这里我们有两个纳什均衡：给出两个玩家都是一个 5 回报的操作 (a1, a1) 和导致一个 10 支付的全局最优 (a2, a2)。

如表 14.2 所示的囚犯困境中，纳什均衡不一定对所有学习器都是最满意的结果。尽管事实是当两人合作的时候两人都将有收获，但在特殊的纳什均衡中两个玩家都更青睐“背叛”动作。合作的结果不具有纳什均衡，这是因为在这个案例中两个玩家都可以通过切换成“背叛”动作来提高他们的奖励。

第一个博弈 (猜硬币) 是不纯粹的策略纳什均衡，因为不纯粹的策略是对另一个纯粹的最佳反应的最佳反应。相反，博弈中的纳什均衡是两个玩家以相同的概率选择对方。也就是说，纳什均衡策略组合是 ((1/2, 1/2), (1/2, 1/2))。

448

14.2.2 重复博弈中的强化学习

上述中的博弈常常可当作多学习器强化学习技术的例子。与博弈论中的理论背景不同，这里不会假设学习器为完全访问支付矩阵。在强化学习的设定中，学习器被当作标准型博弈中的玩家，并反复进行博弈，以便随着时间的推移改进策略。

值得注意的是这些重复博弈还没有捕获所有的多学习器强化学习问题。在重复博弈中所有预期奖励的改变都是源于玩家们的策略改变。学习器外部的环境状态或状态转换函数是没有变化的。因此重复博弈有时也称为无状态博弈。尽管有这些限制，我们还会继续在这一节深入探索，这些博弈已经为独立学习学习器提供一个富有挑战的问题，并且非常适合测试协调方法。在下一节，我们将会介绍包含动态环境的马尔可夫博弈框架。

在处理博弈中的强化学习时，必须考虑一些不同的情况。像 RL 研究中常见的一样，我们假设最开始学习器不知道博弈已经开始，这与传统经济博弈论文献相反。也就是说，学习器没有获得回报函数且不知道由于执行了某个（联合）动作而得到预期回报。然而，RL 技术仍不同于学习器的观察结果。此外，我们还假设博弈奖励是随机的，这意味着一个联合动作并不总是给每个学习器相同确定的回报。因此，不得不简单地重复动作。

由于预期的回报取决于所有学习器的策略，所以许多多学习器 RL 方法会假设学习器能够观察到博弈中所有参与者的动作或回报。以便学习器对其对手进行建模并明确地了解对其联合动作的估计。不过，可以认为这种假设是不现实的，因为在完全分布的多学习器系统中，该信息可能不容易获得。在案例中 RL 技术必须有能力处理由其他学习器引起的非平稳回报。因此，当开发多学习器强化学习应用程序时，重要的是考虑在特定设定中可用的信息，以便将该设定与适当的技术进行匹配。

14.2.2.1 学习的目的

449 由于在博弈中无法同时让所有玩家最大化其收益，所以大多数 RL 方法都是尝试取得纳什均衡。然而，关于纳什均衡的批评也属于这类学习方法。第一个问题是纳什均衡不一定是唯一的，这就导致了均衡选择问题。一般的，在一个博弈中可以存在多个纳什均衡。这些均衡对玩家给出不同的回报。这意味着一个学习纳什均衡的方法，不能保证一个特定的结果，甚至给玩家一个特定的回报。可以从表 14.2c 协调博弈中看见，在这个表里有两个纳什均衡存在，一个将 5 支付给学习器，另一个给了 10 支付。表 14.3b 中的博弈也有多个纳什均衡，而最佳的是 $(a1, a1)$ 或 $(a3, a3)$ 中的一个。这导致了一个学习学习器的协调问题，因为这两个 NE 质量是相同的。

此外，因为即使在平衡博弈中玩家也可以有不同的预期支付，所以不同的玩家也偏爱不同的平衡结果，这意味着应该注意确保玩家在单一均衡上进行协调。这种情况可以在表 14.2d 中的性别之战中观察到，其中存在两个单纯的纳什均衡，但每个玩家却喜欢不同的平衡结果。

另一个批判是纳什均衡不能保证最优。虽然正在进行的纳什均衡可以保证没有单一玩家通过单方面改变其策略来提高他的回报，但并不能保证玩家全局上最大程度地获得收益，或甚至没有玩家同时做得更好的游戏存在。这可能是博弈中有非纳什均衡的结果，尽管如此，这会导致所有学习器的收益高于它们在纳什均衡时收到的收益。这可以从表 14.2c 中的例子看出。

虽然经常作为主要学习目标，但在博弈论中纳什均衡不是唯一可行的解决方案。从某种

意义上来说, 部分因为上述批评已经开发了用于博弈的一些替代解决方案。这些替代方案中包括了其他均衡概念的范围, 比如协作均衡 (CE) [Aumann, 1974], 它概括了纳什均衡的概念; 或稳定进化对策 (ESS) [Smith, 1982], 它改进了纳什均衡。这里每一种均衡结果都有它的应用和优 / 缺点。使用哪个解决方案的概念取决于当前的问题, 以及学习算法的目标。关于可能均衡概念的完整讨论超出了本章的范围。我们注重的是纳什均衡并简要提及遗憾最小化, 因为这些是多学习器学习文献中最常见到的方法。更多的解决方案概念的完整讨论在许多书上都能找到, 比如 [Leyton-Brown and Shoham, 2008]。

在继续之前, 我们提出另一个评估标准, 它经常用于重复博弈: 遗憾概念。遗憾是学习器现实的收益与学习器使用某种固定策略获得的最大收益之间的差异。通常, 将学习器性能与之进行比较的固定策略, 只不过是学习器的纯策略。在这个案例中, 学习器的全部遗憾是所获得的回报与学习器做了某些固定动作而获得的回报之间的累积差异。对于一个学习器 k , 给定在时间 T 进行的历史收益, 这可以定义为:

450

$$\mathcal{R}_T = \max_{a \in A_k} \sum_{t=1}^T R_k(\mathbf{a}_{-k}(t) \cup \{a\}) - R_k(\mathbf{a}(t)) \quad (14.1)$$

这里 $\mathbf{a}(t)$ 表示在时间 t 的联合动作并且 $\mathbf{a}_{-k}(t) \cup \{a\}$ 表示相同的联合动作并随玩家 k 进行的动作 a 。大多数基于学习方法的遗憾都尝试将学习器的平均遗憾 \mathcal{R}_T/T 降至最低。这个遗憾的精确计算需要知道奖励函数和其他学习器的观察动作来决定 $R_k(\mathbf{a}_{-k}(t) \cup \{a\})$ 的项。如果信息无法使用, 遗憾要从以前的观察中估算。在一些假设下遗憾的学习可以看成收敛于某种形式的均衡发挥 [Foster and Young, 2003; Hart and Mas-Colell, 2001]。

14.2.2.2 在博弈中的 Q 学习

我们自然会提问当学习器使用标准的、单一智能体的 RL 技术在博弈环境中相互影响时会发生什么。早期的多学习器 RL 探索注重于大的重复博弈 Q 学习的应用。在这种所谓的独立或不知情的设置中, 每个玩家 k 保持估计 Q 值的个人向量 $Q_k(a)$, $a \in A_k$ 。玩家可以在不需要在博弈中使用其他玩家的任何信息的情况下通过自己的动作来学习 Q 值。由于在重复博弈中没有环境状态的概念, 所以单一的估计向量是足够的, 而不是满表的状态 - 动作对, 且标准的 Q 学习更新通常可简化为无状态版本:

$$Q(a) \leftarrow Q(a) + \alpha[r(t) - Q(a)] \quad (14.2)$$

在动态的重复标准型无状态 Q 学习 [Claus and Boutilier, 1998] 中, 对共同利益博弈进行了研究。其关键问题是: 简单的 Q 学习是否仍保证在多学习器设定中收敛, 如果是, 它是否能收敛到 (最佳) 均衡。它还把独立的 Q 学习器与联合动作学习器 (见下文) 联系起来并查询收敛的速率、被博弈结构和动作选择策略影响的限制点。在一个研究的关系分支 [Tuyt's and Nowé, 2005; Wunder et al, 2010] 中独立 Q 学习是学习来自进化博弈论 [Smith, 1982] 的技术。

虽然独立的 Q 学习器在某些情况下能达到平衡, 但也会在一些博弈中表现出协调不一致, 甚至在其他地方没有完全融合。

拿联合动作学习器和独立学习器做比较。在前者中, 学习器为所有联合动作学习 Q 值, 换句话说, 每个学习器 j 都可以学习 A 中的所有 a 的 Q 值。每个学习器根据学习器对其他学习器的策略进行单独的选择操作。公式 (14.3) 表示联合动作的 Q 值根据其他学习器将选择对某个特定值的概率进行加权。预期值 (EV) 可以用于任何动作选择技术的组合。[Claus

451

and Boutilier,1998] 通过实验 (实验采取表 2 中的博弈) 表明, 使用降低温度的 Boltzmann 探索策略的共同学习器和独立学习学习器表现出的动作非常相似。这些学习器已经从 [Tuyt's and Nowé, 2005] 的进化博弈理论的角度进行了研究, 结果表明这些学习器最终将会收敛于发展稳定的 NE 状态, 这种状态不一定是帕累托最优的。

$$EV(a^i) = \sum_{a^{-i} \in A_{-i}} Q(a^{-i} \cup \{a^i\}) \prod_{j \neq i} \{Pr_{a^{-i}[j]}^i\} \quad (14.3)$$

然而学习器要达到最优 NE 是十分困难的, 并且需要更复杂的探索策略来增加收敛到最优 NE 的概率。简单的探索策略不够充分的原因主要是由于最优 NE 涉及的动作往往会比其他动作相结合的收益低得多。举个例子在表 14-3a 中, 这里当行的玩家的动作 a1 组合列玩家的动作 a1 时将只会导致一个最高回报 11。在学习阶段, 学习器会保持探索并且动作 a1 也会与动作 a2 和 a3 组合。作为结果, 学习器会接受更加“安全”的 NE (a2, a2)。在表 14-3b 中我们也观察到了相似的动作, 因为在 2 NE 中不协调的动作会受到惩罚, 惩罚越大, 博弈就越难达到任何一种 NE 最优。这也解释了为什么当允许独立学习器去使用任何方法 (包括一个随机的探索策略) 时, 通常情况下它们也不能收敛于 NE。然而, 在只含单学习器的 Q 学习中, 这种特殊的探索策略并不会影响最终的收敛 [Tsitsiklis, 1994], 它不再在 MAS 设置中保存。

单学习器 Q 学习的局限性引出了多个重复博弈的 Q 学习算法的扩展。这些方法大多集中于协调机制, 使得 Q 学习器能够在共同利益博弈中达到最佳结果。举个例子, 频率最大的 Q 学习 (FMQ) 算法 [Kapetanakis and Kudenko, 2002] 保持频率值 $freq(R^*, a)$ 代表某个动作 a 已经观察到了迄今为止的最大回报 (R^*) 的频率。然后将这个值用作一系列加入了 Q 值的探索。FMQ 算法不是直接使用 Q 值, 而是依赖于对动作的以下启发式评估:

$$EV(a) = Q(a) + w \cdot freq(R^*, a) \cdot R^* \quad (14.4)$$

这里 w 是权重, 它包含了重要的探索值 $freq(R^*, a)$ 。实验结果表明, 该算法能够使学习器在具有确定收益的共同利益博弈中达到最优的联合动作。

在 [Kapetanakis et al, 2003] 中, 引入了承诺序列的概念, 允许在具有随机收益的博弈中独立学习。承诺序列是学习器承诺选择出的相同动作的时隙。这些时隙的顺序根据一些已知的学习器协议产生。在属于相同顺序的时隙中使用这些保证, 学习器承诺始终选择相同的单独动作, 学习器能区分不确定性的两个来源: 回报的噪音信号和通过在影响其他学习器的动作的回报。使学习器以随机收益来处理博弈。

最近多学习器 Q 学习方法的概述在可见 [Wunder et al, 2010]。

14.2.2.3 梯度上升法

作为著名的 Q 学习算法的一个替代, 随着更新我们会列出一些基于梯度的方法。我们将注重于使用学习机器人 (LA) 加强方案的玩家。学习机器人是相对简单的策略迭代器, 它可以将动作集合 A 中的向量动作的概率 \mathbf{p} 保持在一定范围内。就像常见的 RL, 这些概率是基于环境接收到的反馈得来的更新。虽然初步研究大多数聚焦在 n 臂老虎机的设定中的单机机器人, 但已开发出了 RL 算法使用多机器人在 MDP [Wheeler Jr and Narendra, 1986] 中的学习策略。最常用的学习算法更新策略称为线性奖惩机制, 其更新动作概率如下所述:

$$p_i(t+1) = p_i(t) + \lambda_i b(t) (1 - p_i(t)) - \lambda_2 (1 - r(t)) p_i(t), \text{ if} \quad (14.5)$$

$$a(t) = a_i$$

$$p_i(t+1) = p_j(t) - \lambda_1 r(t) p_j(t) + \lambda_2 (1 - r(t)) \left(\frac{1}{K-1} - p_j(t) \right), \text{if} \quad (14.6)$$

$$a_j \neq a_i$$

$r(t)$ 是在时间 t 和 K 中现有机器人动作的数目接收到的反馈。 λ_1 和 λ_2 是常量, 分别叫作奖励和惩罚参数。根据参数 3 的值, 可以考虑算法的不同变化。当 $\lambda_1 = \lambda_2$ 时, 算法称为线性奖励惩罚 (L_{R-P}) 然而当 $\lambda_1 \gg \lambda_2$ 时它叫作线性奖励 ε 惩罚 ($L_{R-\varepsilon P}$)。如果 $\lambda_2 = 0$, 算法叫作线性迟钝奖励 (L_{R-I})。在这个案例中, λ_1 在一些时候也叫作学习率:

$$p_i(t+1) = p_i(t) + \lambda_1 r(t) (1 - p_i(t)), \text{if} \quad (14.7)$$

$$a(t) = a_i$$

$$p_j(t+1) = p_j(t) - \lambda_1 r(t) p_j(t), \text{if} \quad (14.8)$$

$$a_j \neq a_i$$

这个算法也展示出加强更新规则的特殊例子 [Williams, 1992]。尽管事实是所有的这些更新规则来自相同的策略, 但它们展示了非常不同的学习动作。有趣的是, 即使在博弈中没有其他玩家的任何信息 (动作、回报、策略), 这些学习方案依然能在博弈环境中表现良好。每个学习器独立应用 LA 更新规则以改变其动作的概率。下面我们列出一些在博弈设定中 LA 的有趣的属性。在双人零和博弈中, 在它存在的单纯策略中 L_{R-I} 方案收敛到纳什均衡, 然而 $L_{R-\varepsilon P}$ 方案可以近似混合均衡。在 n 玩家共同利益博弈中回报无效也收敛到了纯粹的纳什均衡。在 [Sastry et al, 1994] 中, 研究了一般和博弈中回报无效的动态。作者用一个常微分方程组逼近机器人的更新。发现以下属性适用于 L_{R-I} 动态研究: [453]

- 所有纳什均衡都是不动点。
- 所有严格的纳什均衡都是逐渐稳定的。
- 所有不是纳什均衡的不动点都不是稳定的。

此外, 在 [Verbeeck, 2004] 中, 显示使用回报反应方案的机器人队伍将收敛到概率为 1 的单纯的联合策略, 以及具有随机支付的冲突利益博弈。这些结果在一般的 n 玩家总和博弈中就意味着局部已经收敛到纯纳什均衡 [Verbeeck, 2004]。因为高报偿的 NE 对 LA 有更强吸引力, 所以学习器可能达到更好的 NE。配备仅需要非常有限的通信的探索策略, 学习器能够探索有趣的 NE 而不需要穷举探索, 并且一旦找到, 就可以考虑不同的解决方案概念, 例如在不同的帕累托最优解之间交替的公平解决方案。

[Verbeeck et al, 2005] 中也已经展示了这些基于 LA 的方法能够在学习器异步地采取动作并且在延迟回报的设定中收敛, 这在负载均衡设定或拥塞博弈中是常见的。

无穷小梯度上升 (IGA) 系列算法是博弈中经常研究的另一种梯度技术 [Singh et al, 2000; Bowling and Veloso, 2001; Zinkevich, 2003; Bowling, 2005]。除了在一些重复博弈设定中展示的纳什均衡收敛之外, 这几篇论文中也讨论了它们算法的不足。

14.3 顺序博弈

传统的博弈理论只考虑了多学习器环境下学习中存在的一些重要问题, 而忽略了学习器强化学习全部的复杂性。强化学习中要解决的一个重要问题是在具有状态转换的环境中顺序执行策略。但是不能使用标准型博弈, 因为它们仅允许静态的、随机的奖励函数。在标准型博弈中没有系统可以进行状态转换并且也没有这个概念。这是马尔可夫决策过程这个概念的 [454]

核心问题。因此，我们现在考虑使用一个概括了重复博弈和马尔可夫决策过程的更丰富的框架。我们引入多个学习器到马尔可夫决策过程模型中。环境中的奖励和状态转换现在都取决于系统中学习器的动作。而且，因为学习器可以有不同的目标，所以最大程度地提高所有学习器的奖励是不存在所谓的最佳解决方案的。

为了适应这个问题不断增加的复杂性，我们使用到了马尔可夫博弈随机指标 [Shapley, 1953]。马尔可夫博弈（最初在博弈论中作为延伸标准化博弈被引用）概括了马尔可夫决策过程，最近又提出可以作为多学习器强化学习的标准框架 [Littman, 1994]。马尔可夫博弈假设状态转换后仍是马尔可夫的，然而，状态转换的概率和预期奖励却都取决于所有学习器的联合动作。马尔可夫博弈可以看作马尔可夫决策过程的拓展和多状态环境中的重复博弈。如果我们假设只有一个学习器，或者其他学习器只采用固定策略时，马尔可夫博弈可以简化为马尔可夫决策过程。当马尔可夫博弈仅具有一个状态时，它会恢复成正常形态的博弈。

14.3.1 马尔可夫博弈

单一学习器马尔可夫决策过程（MDP）可以通过马尔可夫博弈扩展到多学习器的境况。在马尔可夫博弈中，联合动作是多个学习器独立选择动作的结果。

定义 14.4 一个马尔可夫博弈是一个元组 $(n, S, A_1, \dots, A_n, R_1, \dots, R_n, T)$:

- n 为系统中学习器的个数。
- 设 $S = \{s^1, \dots, s^N\}$ 为系统状态的集合。
- A_k 为学习器 k 的动作集合。
- $R_k: S \times A_1 \times \dots \times A_n \times S \rightarrow \mathbb{R}$, k 是学习器的奖励函数^①。
- $T: S \times A_1 \times \dots \times A_n \rightarrow \mu(S)$ 是转换函数。

注意， $A_k(s^i)$ 为学习器 k 在 s^i 状态可以采取的动作的集合，其中 $k: 1 \dots n$, n 是在系统中的学习器的数量, $i: 1, \dots, N$, N 是在系统中的状态数量。转换概率 $T(s^i, \mathbf{a}^i, s^j)$ 和奖励 $R^k(s^i, \mathbf{a}^i, s^j)$ 现在都取决于当前的状态 s^i 、下一个状态 s^j 以及状态 s^i 的联合动作，也就是 $\mathbf{a}^i = (a_1^i, \dots, a_n^i)$, $a_k^i \in A_k(s^i)$ 。奖励函数 $R_k(s^i, \mathbf{a}^i, s^j)$ 现在对于每个学习器 k 都是独立的。不同的学习器可以从相同的状态转换收到不同的奖励。我们再次假设博弈中的状态转换服从马尔可夫性质。

在马尔可夫决策过程的例子中，学习器会尽量优化它的未来预期奖励。一个典型的例子是它们会尝试最大化未来折扣奖励或它们在时间上的平均奖励。与单一学习器 RL 大不相同的是现在这些标准也与学习器的策略有关。这将导致学习器 k 在执行一个联合策略 $\pi = (\pi_1, \dots, \pi_n)$ （它向每一个学习器 i 分配策略 π_i ）时，它的预期折扣回报的定义如下：

$$V_k^\pi(s) = E^\pi \left\{ \sum_{t=0}^{\infty} \gamma^t r_k(t+1) \mid s(0) = s \right\} \quad (14.9)$$

在联合策略下，学习器 k 的平均奖励定义为：

$$J_k^\pi(s) = \lim_{T \rightarrow \infty} \frac{1}{T} E^\pi \left\{ \sum_{t=0}^T r_k(t+1) \mid s(0) = s \right\} \quad (14.10)$$

由于不可能同时使所有学习器的标准都最大化，当学习器的目标有冲突时，学习器在执行马尔可夫博弈时将会面临相互协调的问题。因此，通常它会再次依赖平衡点作为解决这些问题的方法。如果其他学习器采取策略固定，则没有学习器 k 的策略会给出更高的预期未

① 与 MDP 的情况相同，还要考虑等效案例（在这个情况下，奖励不依赖于下一个状态）。

来奖励, 所以最佳响应和纳什均衡概念通过定义策略 π_k 作为最佳响应扩展到马尔可夫博弈中去。

应当注意的是, 在学习策略方面, 在马尔可夫博弈中的学习引入了 MDP 中学习策略的新问题。在一个马尔可夫决策过程中, 最优策略总是存在 (给定的基础假设)。这意味着只考虑将每个状态准确地映射到动作的那些策略就足够了。然而在马尔可夫博弈中只考虑这些却不能保持均衡, 所以我们必须考虑策略之间的均衡问题。与在重复博弈中的解决方案一样, 它可能是已折扣的马尔可夫博弈, 或仅有纳什均衡的随机策略。在这种情况下, 仅让学习器的动作对应到相应的状态是不够的: 它们必须能够学习一个混合的策略。当考虑到其他奖励标准时, 这种解决方案会变成得更加困难, 例如平均奖励, 因为固定策略的平衡是不可能存在的 [Gillette, 1957]。这意味着为了实现平衡, 学习器必须要在整个学习过程中随时调整选择动作的策略。幸运的是, 我们可以在问题的结构中引入一些假设来保持静态平衡 [Sobel, 1971]。

14.3.2 马尔可夫博弈中的强化学习

虽然在标准型博弈中, 强化学习面临的问题主要来自学习器之间的相互作用, 但在马尔可夫博弈中, 其面临的问题则来自额外的状态转换。这意味着学习器需要将协调方法或重复博弈中的均衡解算器与来自单智能体 RL 的马尔可夫决策过程的方法相结合。

456

14.3.2.1 值迭代

为了把 Q 学习算法成功扩展到多学习器系统中, 我们尝试了很多方法。后来, 我们发现算法必须首先处理掉一些关键问题才可以应用到多学习器的系统中。

首先, 即时奖励与状态转换取决于所有学习器的动作。因此, 在一个多学习器 Q 学习方法中, 学习器不会为每个状态中的动作都对估计 $Q(s, \mathbf{a})$, 但是估计 $Q(s, \mathbf{a})$ 会给出在状态 s 中的联合动作 $\mathbf{a} = a_1, \dots, a_n$ 的预期未来奖励。因此, 与单一学习器情况相反的是, 在多学习器环境中学习器没有对在状态 s 执动作 a_k 的预期奖励做出评估。但是, 要保留评估的向量 (动作 a_k 的未来奖励), 取决于其他学习器的联合动作 \mathbf{a}_{-k} 。在学习期间, 学习器选择一个动作后需要其他学习器来观察执行的动作, 以便更新合适的 $Q(s, \mathbf{a})$ 值。

多学习器 Q 学习需解决的第二个问题是: 脱靴法中使用的状态值应该如何更新。在单个学习器 Q 学习的更新规则中学习器使用在下一个状态 s' 中动作值最大的动作。这给出了在贪婪策略之下当前估计出的状态 s' 值。但如上所述, 因为在下一个状态中采取动作的值还取决于其他学习器的动作, 所以学习器无法预测这个值, 为了处理这些问题, 开发出来一个不同的方法, 它通过考虑其他的学习器来计算一个状态 s' 的值。在下面举出的一些例子中, 所有这些算法都符合在算法 23 给出的一般多学习器 Q 学习的模板, 尽管每个算法在 Q 学习更新中用于计算 $V_k(s')$ 的方法不同。

状态 $V_k(s)$ 中最可能的期望值可以使用对手模型来确定。如果学习器能够估计其他学习器使用的策略, 则它可以使用该信息来确定不同联合动作的预期概率。基于这些概率, 学习器可以确定状态的预期值。这就是接下来要讲的方法——联合动作学习器 (JAL) 算法 [Claus and Boutilier, 1998]。联合动作学习器保持每一个状态中动作对 (s, \mathbf{a}_{-k}) 的数目 $c(s, \mathbf{a}_{-k})$ 。然后, 这些信息可以用来确定其他学习器可能采取的联合动作的经验频率:

$$F(s, \mathbf{a}_{-k}) = \frac{c(s, \mathbf{a}_{-k})}{\sum_{\mathbf{a}_{-k}' \in \mathbb{A}_{-k}} n(s, \mathbf{a}_{-k}')}$$

457

```

t=0
 $Q_k(s,a) = 0 \forall s,a,k$ 
repeat
  for all agents k do
    select action  $a_k(t)$ 
  execute joint action  $\mathbf{a} = (a_1, \dots, a_n)$ 
  observe new state  $s'$ , rewards  $r_k$ 
  for all agents k do
     $Q_k(s,\mathbf{a}) = Q_k(s,\mathbf{a}) + \alpha [R_k(s,\mathbf{a}) + \gamma V_k(s') - Q_k(s,\mathbf{a})]$ 
until Termination Condition

```

算法 23 多学习器 Q 学习

其他学习器的估计频率允许当前学习的学习器为一个状态计算预期 Q 值：

$$V_k(s) = \max_{a_k} Q(s, a_k) = \sum_{\mathbf{a}_{-k} \in \mathbb{A}_{-k}} F(s, \mathbf{a}_{-k}) \cdot Q(s, a_k, \mathbf{a}_{-k})$$

$Q(s, a_k, \mathbf{a}_{-k})$ 表示在状态 s 下 Q 值的联合动作，在这个联合动作中智能体 k 运行 a_k 且其他智能体会根据 \mathbf{a}_{-k} 来运行。这些预期 Q 值可以用作学习器动作的选择，和 Q 学习更新一样，只作为在标准的单一学习器 Q 学习的算法。

也可以使用另一个常用在多学习器 Q 学习中的方法，即假设其他学习器将根据某些策略进行展示。比如，在极小极大的 Q 学习算法中 [Littman, 1994]，这个算法在双学习器零和问题上被开发出来，学习学习器假设它的对手将采取最小化学习器收益的动作。这意味着单学习器 Q 学习的最大算子被极大极小值代替：

$$V_k(s) = \min_{a'} \max_{\sigma \in \mu(A)} \sum_{a \in A} \sigma(a) Q(s, a, a')$$

Q 学习学习器最大程度地使用它在状态 s 中的策略，同时假设对手将采用最小化学习器的预期奖励的动作。请注意，学习器不只是在确定策略下最大化，因为最大值有时候还需要一些混合策略。这个系统后来推广到敌我 Q 学习 [Littman, 2001a]，学习学习器通过将它们标记为朋友，来协助最大化其奖励或敌对，并尝试尽量减少收益来处理多个学习器。

假设学习器待选方法会执行一个均衡策略。比如，纳什 Q [Hu and Wellman, 2003] 观察所有学习器的奖励并且为所有学习器（不仅仅包括当前在学习的学习器，还有其他学习器）估计了 Q 值。这使学习器作为一个博弈，表现在每个状态中的联合动作选择中，这里奖励矩阵通过联合动作中学习器的 Q 值定义。这个表示也叫作阶段博弈。

纳什 Q 学习器假设所有的学习器会根据在每一个状态的阶段博弈中的纳什均衡来运行：

$$V_k(s) = \text{Nash}_k(s, Q_1, \dots, Q_n)$$

这里的 $\text{Nash}_k(s, Q_1, \dots, Q_n)$ 表示当学习器随着 Q 值 Q_1, \dots, Q_n 使用在阶段博弈中的纳什均衡时，学习器 k 的预期支付。在一些相当严谨的基于阶段博弈结构上的假设中，纳什 Q 可以显示出在自我对局中收敛到两个学习器策略的纳什均衡。

在纳什 Q 中使用的方法也可以与其他均衡概念相结合，比如与平衡相关的 [Greenwald et al, 2003] 或斯塔科尔伯格均衡 [Kononen, 2003]。这些方法最大的不同是当多个均衡存在时值没有被特殊定义，并且协调需要取得相同的均衡。在这些案例中，通常需要额外的机制来选择一些平衡。

虽然在基于多学习器 RL 的值迭代的加强研究中有了一些理论保证 [Littman, 2001b]，但是一般的马尔可夫博弈案例的收敛结果依旧难懂。此外，最近研究表明，仅依靠 Q 值可能不足以在任意一般和博弈中学习均衡策略 [Zinkevich et al, 2006]，所以需要新的方法。

14.3.2.2 策略迭代

在这一节中我们解释多学习器强化学习的策略迭代。我们着重介绍一个叫作马尔可夫博弈的连通学习机器人算法 (MG-ILA)[Vrancx et al, 2008b], 它基于 14.2.2.3 节的学习自动机并应用于平均奖励的马尔可夫博弈。这个算法可以看作一个现实的评估器, 这里使用学习自动机存储策略。大部分的思路都比较直白: 就是每一个学习器 k 把单一的学习自动机 $LA(k, i)$ 放置在每一个状态系统 s 中。在每一次运行时只有当前状态的自动机是运行的。导致联合动作触发下一个状态转换和即时奖励。自动机不使用即时奖励而是使用估计平均奖励的响应来进行更新。完整的算法如算法 24 所示。

这个算法有趣地方是它的限制动作可以近似考虑成一个标准型博弈, 在这个博弈中所有汇总的自动机都是玩家。在博弈中为在每个状态中的每个学习器选择一个动作, 并由此为所有的学习器寻找对应纯粹的联合策略。博弈中的奖励是其对应的联合策略的预期平均奖励。在 [Vrancx et al, 2008b] 中, 结果表明该算法将收敛到在结果博弈中的纯纳什均衡 (如果均衡存在), 并且该均衡对应于学习器策略之间的纯均衡。博弈近似值也使得对学习动态的进化博弈论的分析 [Vrancx et al, 2008a], 这类似于重复博弈的应用。

459

```

initialise  $r_{prev}(s, k), t_{prev}(s), a_{prev}(s, k), t, r_{tot}(k), \rho_k(s, a), \eta_k(s, a)$  to zero,  $\forall s, k, a$ .
 $s \leftarrow s(0)$ 
loop
  for all Agents  $k$  do
    if  $s$  was visited before then
      • Calculate received reward and time passed since last visit to state  $s$ :


$$\Delta r_k = r_{tot}(k) - r_{prev}(s, k)$$


$$\Delta t = t - t_{prev}(s)$$


      • Update estimates for action  $a_{prev}(s, k)$  taken on last visit to  $s$ :


$$\rho_k(s, a_{prev}(s, k)) = \rho_k(s, a_{prev}(s, k)) + \Delta r_k$$


$$\eta_k(s, a_{prev}(s, k)) = \eta_k(s, a_{prev}(s, k)) + \Delta t$$


      • Calculate feedback:


$$\beta_k(t) = \frac{\rho_k(s, a_{prev}(s, k))}{\eta_k(s, a_{prev}(s, k))}$$


      • Update automaton  $LA(s, k)$  using  $L_{R-I}$  update with  $a(t) = a_{prev}(s, k)$  and  $\beta_k(t)$  as above.
      • Let  $LA(s, k)$  select an action  $a_k$ .
      • Store data for current state visit:


$$t_{prev}(s) \leftarrow t$$


$$r_{prev}(s, k) \leftarrow r_{tot}(k)$$


$$a_{prev}(s, k) \leftarrow a_k$$


      • Execute joint action  $\mathbf{a} = (a_1, \dots, a_n)$ , observe immediate rewards  $r_k$  and new state  $s'$ 
      •  $s \leftarrow s'$ 
      •  $r_{tot}(k) \leftarrow r_{tot}(k) + r_k$ 
      •  $t \leftarrow t + 1$ 

```

算法 24 MG-ILA

虽然不像基于值迭代的方法那样普及, 但是已经提出了一些基于策略迭代的有趣

的方法。与上述算法一样，这些方法通常依赖于一个基于策略空间搜索的梯度。例如，[Bowling and Veloso, 2002]，提出一个评估器结合 tile 编码泛化和策略梯度上升的框架并使用变学习率学习（WoLF）法则。由此算法可经验性地在其他的大问题上应用于学习。[Kononen, 2004] 引入共同利益马尔可夫博弈的策略梯度方法，其扩展了 [Sutton et al, 2000] 提出的单一学习器的方法。最后，[Peshkin et al, 2000] 开发了一种基于梯度的策略搜索方法，以用于局部观察和相同支付随机博弈。已证明该方法收敛到局部最佳，而这里的局部最佳不一定是学习器策略之间的纳什均衡。

14.4 在多学习器系统中的稀疏交互

在马尔可夫博弈中强化学习需要注意的是当学习器学习时状态动作空间的大小。所有学习器都在联合状态空间中学习，因此，这些方法对于除了最小的环境和有数量限制以外的学习器都会失效。最近，我们为了缓和这个问题花了许多精力。这些方法的主要思想是只有通过这样做才能获得更好的收益，才能明确地考虑其他学习器。在任何其他情况下，可以安全地忽略其他学习器，这在小型状态-动作空间中学习是有益的，同时还可以访问必要的信息来处理其他学习器存在的问题。一个相同系统的例子是自动化仓库，在这里自动化引导车只有当它们被其他人关闭时才考虑其他学习器。我们可以分成两个不同的研究路线：学习器可以根据所选择的动作决定协调策略和在何时观察其他学习器的状态信息对学习有利。我们将分别在 14.4.2.1 节和 14.4.2.2 节中描述这些方法。

14.4.1 多等级学习

稀疏交互学习提供了一种简单的方法来处理在学习过程中涉及的学习器状态空间的指数级增长。学习器应该只取决于更多全局信息，在学习器状态的转换对应下一个状态或奖励的情况下，学习器经验不仅依赖于局部状态信息，而且还取决于在状态中的信息或其他学习器的动作。稀疏交互的思维是“什么时候一个学习器受到另一学习器的影响？”回答这个问题，要使一个学习器知道什么时候它可以选择独立动作（也就是说状态转换方法对多学习器的联合动作有一定的影响），这通常导致一个多学习器学习进程分离成两个分开的图层。顶层需要学习何时来观察其他学习器的状态信息，并选择是只需单纯地独立学习还是也需对学习器进行协调。底层包含了一个单一的学习器学习技术（当没有其他主体的影响时，这种技术会被使用），和一个多学习器技术（当主体接收到的状态转换和奖励信息独立于当前的状态和其他主体的动作时，这种技术会被使用）。图 14.2 展示了这个框架的图解。在下文中我们首先介绍算法的概述（这个概述是处理这个来自视图的动作空间点的问题，并且专注在动作的协调上）。

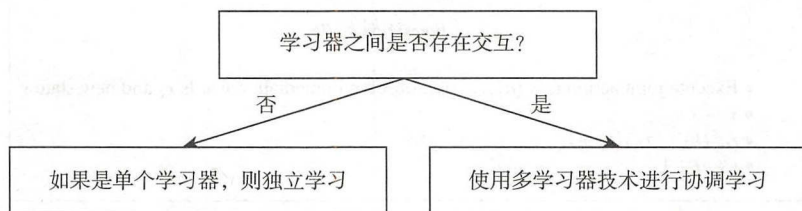


图 14.2 何时一个学习器受另一个学习器影响的图示

14.4.2 协调学习与稀疏交互

14.4.2.1 在学习器之间学习相互依赖

Kok 和 Vlassis 提出了一种基于学习器联合动作空间的稀疏表示方法,可以同时观察整个联合状态空间。更具体地说,他们对学习这些状态的联合动作值感兴趣,这里学习器明确需要协调。在许多问题中,这些需求只是发生在非常特殊的情况下 [Guestrin et al, 2002b]。稀疏表格多学习器 Q 学习在必要协调时会保存许多的状态。在这些状态中学习器选择一个联合动作,而在所有不协调状态中它们都是各自选择一个动作 [Kok and Vlassis, 2004b]。通过用协调图 (CG) 取代这些状态表,可以表现为被少数学习器限制的依赖关系 [Guestrin et al, 2002a; Kok and Vlassis, 2004a, 2006]。这些技术称为稀疏合作 Q 学习 (SCQ)。图 14.3 展示了一个简单 CG 的图解表示法描述的一种情况,这里学习器 4 的动作影响依赖学习器 2 并且学习器 2 和学习器 3 的动作都依赖学习器 1 的动作,这里节点表示学习器,而边定义了两个学习器之间的依赖关系。如果学习器转换到协调状态,它们实行一个计算当前状态的最优联合动作的变量消除算法。在所有其他状态中,学习器都会独立选择它们的动作。

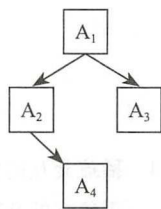


图 14.3 简单协调图。在这情况中描述学习器 4 的动作影响依赖学习器 2 且学习器 2、3 都依赖于学习器 1

在下面的工作中,我们会介绍有益协调 [Kok et al, 2005]。这是一个更先进的算法,它的使用和 SCQ 的思路一样,但不是使用已经事先定义好的 CG,它们在线学习。这是通过维护关于其他学习器的状态和动作获得的奖励的统计信息来完成的。因此,有可能学习学习器之间存在的上下文特定依赖性并可以在 CG 中表示它们。然而,这种技术限于完全协作的 MAS。

这些方法的首要目标是减少联合动作空间。然而,如上所述的算法中的计算或者学习,通常采用完整的联合状态空间中多学习器视角去选择它们的动作,即使在只使用局部状态信息的状态中也是充分的。因此,它们正在学习的状态空间在学习器数量方面仍然是指数级的并且其仅限于在可以观察整个联合状态的情况下使用。

14.4.2.2 学习更丰富的状态空间表示法

新的方法是学习有利于包含其他学习器状态信息的环境状态,而不是明确地学习最佳的协调动作。我们将阐述两个不同的方法。第一个方法学习在状态协调中有利于使用 RL 方法。第二个方法是一套基于在必要的状态中观察奖励协调的学习状态。不像 14.4.2.1 节中所提到的方法那样,这些方法也可以应用于冲突利益博弈且允许独立动作选择。

在这一节中方法所阐述的大意如图 14.4 所示。为了避免次优的奖励,这些算法将扩展到学习器的局部状态信息,以用来合并另一个学习器的信息(如果这些信息是必要的)。

1. 协调学习

[Spaan and Melo, 2008] 从一个与 Kok 和 Vlassis 不同的角度处理协调问题。对于不确定条件下的多学习器决策,其引入了一个新的模型,叫作合作驱动马尔可夫博弈 (IDMG)。这个模型包括一套互动状态,当协调有利时它能列出所有的状态。

在后面的研究中,[Melo and Veloso, 2009] 介绍一个算法,学习器在需要制约它们在其他学习器的局部状态信息的动作的状态下学习。同样的,它们的方法可以看成解决 IDMG

的方法, 注意这里需要协调的状态不是事先标明的。为了实现这个目的, 它们用伪协调动作 (COORDINATE) 增大了每一个学习器的动作空间。这个动作会执行主动感知步骤。例如, 这可以通过向学习器广播以泄露其位置或使用相机或传感器来检测其他学习器的位置。主动感知步骤将决定协调是否有必要或者说如果它是安全的则忽略其他的学习器。因为不协调的惩罚是大于使用主动感知的成本的, 学习器学习在潜在 IDMG 的互动状态中去执行这个动作。这种方法通过将其推迟到主动感知机制来解决协调问题。

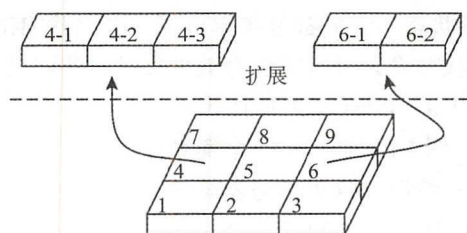


图 14.4 稀疏交互的状态扩展图解表示法。独立单一状态扩展到必要的连接状态。学习器开始于独立的 9 状态。一段时间后, 状态 4 和状态 6 的学习器被扩展为包含另一个状态的状态学习器

LoC 的主动感知步骤可以包括照相机的使用、传感器的数据、暴露在另一个学习器局部状态信息的通信。同样, 算法的结果依赖方法的结果。给定一个主动感知方法, 当协调存在时 LoC 能够学习一系列稀疏状态。注意依赖主动感知方法, 这个算法可以作为冲突利益系统作用于协调之间。

作者使用一个标准 Q 学习更新规则的变体:

$$Q_k^C(s, a_k) \leftarrow (1 - \alpha(t)) Q_k^C(s, a) + \alpha(t) \left[r_k + \gamma \max_{a'} Q_k(s', a'_k) \right] \quad (14.11)$$

这里 Q_k^C 代表当学习器 k 配合并包含用于其独立状态的状态 - 动作值时 Q 表所包含的状态 Q_k 。联合状态信息使用 s 表示, 然而 s_k 和 a_k 是局部状态信息和学习器 k 的动作。所以 Q_k^C 的更新使用 Q_k 的评估。这表示 COORDINATE (协调) 动作的一步动作, 并且允许 Q_k^C 的稀疏表示, 因此在该联合 Q 表中的状态之间没有直接依赖性。

2. 协调 Q 学习

协调 Q 学习 / CQ 学习, 要知道状态学习学习器在哪些状态应该考虑其他学习器 [De Hauwere et al, 2010], 并且其中状态不能仅使用其自己的状态信息。更确切地说, 这个算法会识别状态, 在这个状态中一个学习器应该关闭它的偏爱动作并考虑其他学习器的动作。

算法可以被分解为三个部分: 检验冲突情况, 选择动作和更新 Q 值, 下面将会详细解释。

(1) 检验冲突情况

学习器必须确定哪些状态受到至少一个其他学习器的影响。CQ 学习需要对此有一个基线, 因此, 假设学习器已经学习了一种在特定状态下选择适用单独策略的动作的预期收益模型。比如, 在网络世界中这将意味着学习器已经学会了完成一些目标的策略, 同时是存在于环境的唯一的学习器。如果学习器之间相互影响, 这说明学习器在一起动作的时候会收到奖励。CQ 学习使用统计测试来检测所选择的状态 - 动作对所观察到的奖励是否与其在环境中单独动作的情况相比有变化。可能发生两种情况

- 统计数据可以检测所收到的即时奖励的变化。在这种情况下，算法将标记该状态，并通过从联合状态空间收集新的样本来搜索该变化的原因，以便识别发生碰撞的联合状态-动作对。这些状态-动作对会被标记为危险，并通过添加这种联合状态信息来扩展学习器的状态空间。不引起交互的状态-动作对被标记为安全，也就是说，在这个状态中学习器的动作是来自其他独立学习器的状态。因此，该算法将首先尝试根据其自己的状态来检测学习器收到的奖励的变化，然后再尝试识别由于哪些其他学习器发生这些更改。
- 统计表明学习器收到的回报是来自相同的分布，好像学习器是单独运行的一样。因此，在这种情况下没有采用特殊动作且学习器像只有它自己一样继续运行。

(2) 选择动作

如果学习器选择一个动作，它会检测当前局部状态是否符合以前检验的状态（上述第一种情况）。如果是这样，它将观察全局状态信息，以确定其他学习器的状态信息是否与检测到冲突时相同。如果是这种情况，它会对其全局状态信息进行调整，否则它可以独立运行，只使用自己的本地状态信息。如果它的全局状态信息没有导致偏差（上述第二种情况），它可以在不考虑其他学习器的情况下进行。

465

(3) 更新 Q 值

更新 Q 值遵循与上述协调算法学习相同的想法。用于本地状态的 Q 值可用于引导已扩充的状态的 Q 值。

统计检验在这算法中使用的是学生 t 检验 [Stevens, J.P., 1990]。该测试可以确定在空假设中两个样本群的平均值是否是相等的，还是不相等的。在 CQ 学习中，此测试首先用于识别在这种状态下，观察到的奖励与基于单一学习器学习的预期奖励有显著差异，并且还确定这些变化所依赖的其他学习器的状态。

该算法的形式描述如算法 25 所示。

```

Initialise  $Q_k$  through single agent learning and  $Q_k^j$ ;
while true do
  if state  $s_k$  of Agent  $k$  is unmarked then
    Select  $a_k$  for Agent  $k$  from  $Q_k$ 
  else
    if the joint state information  $js$  is safe then
      Select  $a_k$  for Agent  $k$  from  $Q_k$ 
    else
      Select  $a_k$  for Agent  $k$  from  $Q_k^j$  based on the joint state information  $js$ 
    Sample  $\langle s_k, a_k, r_k \rangle$ 
    if t-test detects difference in observed rewards vs expected rewards for  $\langle s_k, a_k \rangle$  then
      mark  $s_k$ 
      for  $\forall$  other state information present in the joint state  $js$  do
        if t-test detects difference between independent state  $s_k$  and joint state  $js$  then
          add  $js$  to  $Q_k^j$ 
          mark  $js$  as dangerous
        else
          mark  $js$  as safe
    if  $s_k$  is unmarked for Agent  $k$  or  $js$  is safe then
      No need to update  $Q_k(s_k)$ .
    else
      Update  $Q_k^j(js, a_k) \leftarrow (1 - \alpha_t)Q_k^j(js, a_k) + \alpha_t[r(js, a_k) + \gamma \max_a Q(s'_k, a)]$ 

```

算法 25 学习器 k 的 CQ 学习算法

CQ 学习也常用于状态中的概念信息, 以获得学习器之间存在的协调依赖性的状态无关表示 [De Hauwere et al, 2010]。这个信息可以迁移到其他更复杂的作业环境 [Vrancx et al, 2011]。迁移学习的这个原则提高了学习速度, 因为学习器可以仅仅关注手头问题的核心任务, 并使用迁移的经验来协调问题。

这种方法后来扩展到检测稀疏交互, 接下来的几个时间步长只反映在回报信号, [De Hauwere et al, 2011]。例如, 货物到达仓库的顺序是否很重要。

14.5 延伸阅读

多学习器强化学习正往各个研究领域发展, 但是很多具有挑战性的研究问题仍然没有攻克。在单学习器强化学习中已经开展了大量研究工作, 比如贝叶斯强化学习、批处理学习或迁移学习, 但尚未找到多学习器共同体强化学习的方法。多学习器系统的一般概述已经由 [Weiss, G., 1999; Wooldridge, M., 2002; Shoham, Y. and Leyton-Brown, K., 2009] 给出。[Gintis, H., 2000] 为博弈论领域完整的概述做出了很大贡献。

若关注于多学习器强化学习中, 我们推荐阅读 [Busoniu et al, 2008], 它对最近的研究做出了广义上的概括, 并在细节上阐述了一个 MARL 算法的代表性选择以及它们的优势和缺点, 多学习器研究的另一个方向是考虑一个学习器不知道系统中其他学习器的类型或能力的系统 [Chalkiadakis and Boutilier, 2008]。

一个在多学习器强化学习和单学习器强化学习都会遇到的问题样是, 奖励会发生延迟。这样的情况在包含队列的系统中最为典型, 比如网络路由和作业调度。采取动作的即时奖励只能在动作的效果变得明显之后返回给学习器, 例如, 处理作业之后。在 [Verbeeck et al, 2005] 中, 给定了学习机器人策略迭代方法, 而这种延迟回报则是必然的。在 [Littman and Boyan, 1993] 中, 描述了一个基于算法的网络路由值迭代问题。[Steenhaut et al, 1997; Fakir, 2004] 提出了该算法的改进版, 其允许一方面以更有效的方式利用奖励信息, 另一方面又能避免由于不小心的探索而导致的不稳定性问题。

参考文献

- Aumann, R.: Subjectivity and Correlation in Randomized Strategies. *Journal of Mathematical Economics* 1(1), 67–96 (1974)
- Bowling, M.: Convergence and No-Regret in Multiagent Learning. In: *Advances in Neural Information Processing Systems 17 (NIPS)*, pp. 209–216 (2005)
- Bowling, M., Veloso, M.: Convergence of Gradient Dynamics with a Variable Learning Rate. In: *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pp. 27–34 (2001)
- Bowling, M., Veloso, M.: Scalable Learning in Stochastic Games. In: *AAAI Workshop on Game Theoretic and Decision Theoretic Agents* (2002)
- Busoniu, L., Babuska, R., De Schutter, B.: A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 38(2), 156–172 (2008)
- Chalkiadakis, G., Boutilier, C.: Sequential Decision Making in Repeated Coalition Formation under Uncertainty. In: Parkes, P.M., Parsons (eds.) *Proceedings of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, pp. 347–354 (2008)
- Claus, C., Boutilier, C.: The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems. In: *Proceedings of the National Conference on Artificial Intelligence*, pp. 746–752. John Wiley & Sons Ltd. (1998)

- De Hauwere, Y.M., Vrancx, P., Nowé, A.: Learning Multi-Agent State Space Representations. In: Proceedings of the 9th International Conference on Autonomous Agents and Multi-Agent Systems, Toronto, Canada, pp. 715–722 (2010)
- De Hauwere, Y.M., Vrancx, P., Nowé, A.: Detecting and Solving Future Multi-Agent Interactions. In: Proceedings of the AAMAS Workshop on Adaptive and Learning Agents, Taipei, Taiwan, pp. 45–52 (2011)
- Dorigo, M., Stützle, T.: Ant Colony Optimization. Bradford Company, MA (2004)
- Fakir, M.: Resource Optimization Methods for Telecommunication Networks. PhD thesis, Department of Electronics and Informatics, Vrije Universiteit Brussel, Belgium (2004)
- Foster, D., Young, H.: Regret Testing: A Simple Payoff-based Procedure for Learning Nash Equilibrium. University of Pennsylvania and Johns Hopkins University, Mimeo (2003)
- Gillette, D.: Stochastic Games with Zero Stop Probabilities. *Ann. Math. Stud.* 39, 178–187 (1957)
- Gintis, H.: Game Theory Evolving. Princeton University Press (2000)
- Greenwald, A., Hall, K., Serrano, R.: Correlated Q-learning. In: Proceedings of the Twentieth International Conference on Machine Learning, pp. 242–249 (2003)
- Guestrin, C., Lagoudakis, M., Parr, R.: Coordinated Reinforcement Learning. In: Proceedings of the 19th International Conference on Machine Learning, pp. 227–234 (2002a)
- Guestrin, C., Venkataraman, S., Koller, D.: Context-Specific Multiagent Coordination and Planning with Factored MDPs. In: 18th National Conference on Artificial Intelligence, pp. 253–259. American Association for Artificial Intelligence, Menlo Park (2002b)
- Hart, S., Mas-Colell, A.: A Reinforcement Procedure Leading to Correlated Equilibrium. *Economic Essays: A Festschrift for Werner Hildenbrand*, 181–200 (2001)
- Hu, J., Wellman, M.: Nash Q-learning for General-Sum Stochastic Games. *The Journal of Machine Learning Research* 4, 1039–1069 (2003)
- Kapetanakis, S., Kudenko, D.: Reinforcement Learning of Coordination in Cooperative Multi-Agent Systems. In: Proceedings of the National Conference on Artificial Intelligence, pp. 326–331. AAAI Press, MIT Press, Menlo Park, Cambridge (2002)
- Kapetanakis, S., Kudenko, D., Strens, M.: Learning to Coordinate Using Commitment Sequences in Cooperative Multiagent-Systems. In: Proceedings of the Third Symposium on Adaptive Agents and Multi-agent Systems (AAMAS-2003), p. 2004 (2003)
- Kok, J., Vlassis, N.: Sparse Cooperative Q-learning. In: Proceedings of the 21st International Conference on Machine Learning. ACM, New York (2004a)
- Kok, J., Vlassis, N.: Sparse Tabular Multiagent Q-learning. In: Proceedings of the 13th Benelux Conference on Machine Learning, Benelearn (2004b)
- Kok, J., Vlassis, N.: Collaborative Multiagent Reinforcement Learning by Payoff Propagation. *Journal of Machine Learning Research* 7, 1789–1828 (2006)
- Kok, J., 't Hoen, P., Bakker, B., Vlassis, N.: Utile Coordination: Learning Interdependencies among Cooperative Agents. In: Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG 2005), pp. 29–36 (2005)
- Kononen, V.: Asymmetric Multiagent Reinforcement Learning. In: IEEE/WIC International Conference on Intelligent Agent Technology (IAT 2003), pp. 336–342 (2003)
- Könönen, V.: Policy Gradient Method for Team Markov Games. In: Yang, Z.R., Yin, H., Everson, R.M. (eds.) IDEAL 2004. LNCS, vol. 3177, pp. 733–739. Springer, Heidelberg (2004)
- Leyton-Brown, K., Shoham, Y.: Essentials of Game Theory: A Concise Multidisciplinary Introduction. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 2(1), 1–88 (2008)
- Littman, M.: Markov Games as a Framework for Multi-Agent Reinforcement Learning. In: Proceedings of the Eleventh International Conference on Machine Learning, pp. 157–163. Morgan Kaufmann (1994)
- Littman, M.: Friend-or-Foe Q-learning in General-Sum Games. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 322–328. Morgan Kaufmann (2001a)
- Littman, M.: Value-function Reinforcement Learning in Markov Games. *Cognitive Systems Research* 2(1), 55–66 (2001b), <http://www.sciencedirect.com/science/>

- article/B6W6C-430G1TK-4/2/822caf1574be32ae91adf15de90becc4, doi:10.1016/S1389-0417(01)00015-8
- Littman, M., Boyan, J.: A Distributed Reinforcement Learning Scheme for Network Routing. In: *Proceedings of the 1993 International Workshop on Applications of Neural Networks to Telecommunications*, pp. 45–51. Erlbaum (1993)
- Mariano, C., Morales, E.: DQL: A New Updating Strategy for Reinforcement Learning Based on Q-Learning. In: Flach, P.A., De Raedt, L. (eds.) *ECML 2001. LNCS (LNAI)*, vol. 2167, pp. 324–335. Springer, Heidelberg (2001)
- Melo, F., Veloso, M.: Learning of Coordination: Exploiting Sparse Interactions in Multiagent Systems. In: *Proceedings of the 8th International Conference on Autonomous Agents and Multi-Agent Systems*, pp. 773–780 (2009)
- Nash, J.: Equilibrium Points in n -Person Games. *Proceedings of the National Academy of Sciences of the United States of America*, 48–49 (1950)
- Peshkin, L., Kim, K., Meuleau, N., Kaelbling, L.: Learning to Cooperate via Policy Search. In: *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, UAI 2000*, pp. 489–496. Morgan Kaufmann Publishers Inc., San Francisco (2000), <http://portal.acm.org/citation.cfm?id=647234.719893>
- Russell, S., Norvig, P.: *Artificial Intelligence: A Modern Approach*, 2nd edn. Prentice-Hall, Englewood Cliffs (2003)
- Sastry, P., Phansalkar, V., Thathachar, M.: Decentralized Learning of Nash Equilibria in Multi-Person Stochastic Games with Incomplete Information. *IEEE Transactions on Systems, Man and Cybernetics* 24(5), 769–777 (1994)
- Shapley, L.: Stochastic Games. *Proceedings of the National Academy of Sciences* 39(10), 1095–1100 (1953)
- Shoham, Y., Leyton-Brown, K.: *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press (2009)
- Singh, S., Kearns, M., Mansour, Y.: Nash Convergence of Gradient Dynamics in General-Sum Games. In: *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 541–548 (2000)
- Smith, J.: *Evolution and the Theory of Games*. Cambridge Univ. Press (1982)
- Sobel, M.: Noncooperative Stochastic Games. *The Annals of Mathematical Statistics* 42(6), 1930–1935 (1971)
- Spaan, M., Melo, F.: Interaction-Driven Markov Games for Decentralized Multiagent Planning under Uncertainty. In: *Proceedings of the 7th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 525–532. International Foundation for Autonomous Agents and Multiagent Systems (2008)
- Steenhaut, K., Nowe, A., Fakir, M., Dirkx, E.: Towards a Hardware Implementation of Reinforcement Learning for Call Admission Control in Networks for Integrated Services. In: *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications*, vol. 3, p. 63. Lawrence Erlbaum (1997)
- Stevens, J.P.: *Intermediate Statistics: A Modern Approach*. Lawrence Erlbaum (1990)
- Sutton, R., McAllester, D., Singh, S., Mansour, Y.: Policy Gradient Methods for Reinforcement Learning with Function Approximation. In: *Advances in Neural Information Processing Systems*, vol. 12(22) (2000)
- Tsitsiklis, J.: Asynchronous stochastic approximation and Q-learning. *Machine Learning* 16(3), 185–202 (1994)
- Tuyts, K., Nowé, A.: Evolutionary Game Theory and Multi-Agent Reinforcement Learning. *The Knowledge Engineering Review* 20(01), 63–90 (2005)
- Verbeeck, K.: *Coordinated Exploration in Multi-Agent Reinforcement Learning*. PhD thesis, Computational Modeling Lab, Vrije Universiteit Brussel, Belgium (2004)
- Verbeeck, K., Nowe, A., Tuyts, K.: Coordinated Exploration in Multi-Agent Reinforcement Learning: An Application to Loadbalancing. In: *Proceedings of the 4th International Conference on Autonomous Agents and Multi-Agent Systems* (2005)
- Vrancx, P., Tuyts, K., Westra, R.: Switching Dynamics of Multi-Agent Learning. In: *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, vol. 1, pp. 307–313. International Foundation for Autonomous

- Agents and Multiagent Systems, Richland (2008a),
<http://portal.acm.org/citation.cfm?id=1402383.1402430>
- Vrancx, P., Verbeeck, K., Nowe, A.: Decentralized Learning in Markov Games. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 38(4), 976–981 (2008b)
- Vrancx, P., De Hauwere, Y.M., Nowé, A.: Transfer learning for Multi-Agent Coordination. In: *Proceedings of the 3th International Conference on Agents and Artificial Intelligence*, Rome, Italy, pp. 263–272 (2011)
- Weiss, G.: *Multiagent Systems, A Modern Approach to Distributed Artificial Intelligence*. The MIT Press (1999)
- Wheeler Jr., R., Narendra, K.: Decentralized Learning in Finite Markov Chains. *IEEE Transactions on Automatic Control* 31(6), 519–526 (1986)
- Williams, R.: Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 8(3), 229–256 (1992)
- Wooldridge, M.: *An Introduction to Multi Agent Systems*. John Wiley and Sons Ltd. (2002)
- Wunder, M., Littman, M., Babes, M.: Classes of Multiagent Q-learning Dynamics with epsilon-greedy Exploration. In: *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, pp. 1167–1174 (2010)
- Zinkevich, M.: Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In: *Machine Learning International Conference*, vol. 20(2), p. 928 (2003)
- Zinkevich, M., Greenwald, A., Littman, M.: Cyclic equilibria in Markov games. In: *Advances in Neural Information Processing Systems*, vol. 18, p. 1641 (2006)

去中心化的部分可观察马尔可夫决策过程

Frans A. Oliehoek

摘要

本章概述了去中心化 POMDP (Dec-POMDP) 框架。在 Dec-POMDP 中, 一组学习器仅在局部信息的基础上实现全部奖励最大化, 这意味着学习器在执行过程中没有观察到马尔可夫信号, 因此学习器的个别策略只是从历史映射到动作。寻找一个最理想的联合策略是一个非常难的问题: 它是 NEXP 完全的。这表明, 假设 $NEXP \neq EXP$, 任意一个最优的解决方案在最坏的情况下将需要成倍指数幂时间。本章关注于在有限域的情况下规划 Dec-POMDP, 它包含求解 Dec-POMDP 的前向启发式搜索方法以及反向动态规划法, 也讨论了这些方法如何与一个 Dec-POMDP 的最优 Q 值函数相联系, 最后还提供了一些针对其他解决方案和相关深层次话题的建议。

15.1 简介

前面的章节概述了多学习器决策 (第 14 章) 以及不确定状态下 (例如在 POMDP 中 (第 12 章)) 学习器的决策动作。本章进一步考虑了两种不确定状态以及多学习器下的决策, 它尤其关注协作的学习器团体: 这些学习器共享一个单一的对象, 这样的模型能被 Dec-POMDP 框架 [Bernstein et al, 2002] 或大致相同的多学习器决策问题 [Pynadath and Tambe, 2002] 形式化。这一模型的基础构思如图 15.1 所示, 该图描绘了两个学习器做决策的例子。在每一个阶段, 学习器都独立地完成一个动作。环境会发生一个状态转换, 还产生一个取决于所有学习器状态和动作的奖励。最后, 每个学习器接受一个新状态的个别观测值。

这种框架允许为前面章节中介绍的还没实现的模型的重要的真实世界任务建模。一个例子就是队列之间的负载平衡 [Cogill et al, 2004]。每一个学习器用一个队列来代表一个处理单元, 这个队列用于决定是接受新工作还是将工作传给其他队列, 仅基于队列自身大小及其邻近队列大小的局部观测。Dec-POMDP 的另一个重要应用领域是通信网络。例如, 考虑一个数据包的路由任务, 学习器是一个路由器, 它决定每个平均时间步内给哪个相邻节点发送数据包来使包的平均发送时间达到最小 [Peshkin, 2001]。在 Dec-POMDP 社区中得到很多关注的应用程序领域是它的传感器网络 [Nair et al, 2005 ;Varakantham et al, 2007 ;Kumar and Zilberstein, 2009]。其他大家感兴趣的应用领域就是机器人学习器团队 [Becker et al 2004b;Emery-Montemerlo et al, 2005 ;Seuken and Zilberstein, 2007a] 和危机管理 [Nair et al, 2003a,

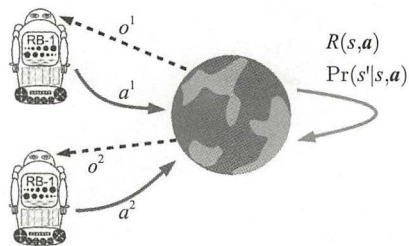


图 15.1 去中心化部分可观察马尔可夫决策过程的执行过程

b;Paquet et al, 2005]。

大多数关于部分可观察多学习器系统的研究都是相对较新的，而且几乎只关注规划（即环境的模型已经给出）而不是建立完全的强化学习 (RL) 模型。本章也只关注于规划。一些 RL 方法在本章的末尾给出。

一个常见的假设是：规划发生在离线阶段，之后计划再被在线执行。这个在线的阶段是完全分散式的，如图 15.1 所示，每个学习器接受在规划阶段发现的联合策略的个别部分[⊖]，以及动作和观测值的个别历史。然而，离线规划阶段是集中式的。我们猜测：单个计算机先计算联合规划，随后将其分配给学习器（这些学习器几乎只需在线执行规划）[⊖]。

472

15.2 Dec-POMDP 框架

在本节，我们将更正式地介绍 Dec-POMDP 模型。我们从给其组成部分的数学定义开始。

定义 15.1 (Dec-POMDP) 一个去中心化的部分可观察马尔可夫决策过程可定义为一个 9 元组 $\langle \mathcal{D}, S, \mathcal{A}, T, R, \mathcal{O}, O, h, I \rangle$ 。其中，

- $\mathcal{D} = \{1, \dots, n\}$ 是 n 个学习器的集合
- S 是环境状态 s 的有限集
- \mathcal{A} 是联合动作有限集
- T 是转换概率函数
- R 是立即奖励函数
- \mathcal{O} 是联合观测有限集
- O 是观察概率函数
- h 是问题的域
- $I \in \mathcal{P}(\mathcal{S})$ 是当 $t=0$ 时的初始状态分布

Dec-POMDP 模型通过考虑联合动作和观测值扩展了单学习器 POMDP 模型，尤其是 $\mathcal{A} = \times_{i \in \mathcal{D}} \mathcal{A}^i$ 是一个联合动作集，这里， \mathcal{A}^i 指学习器 i 可能的动作集，每一个学习器的动作集都不相同，每一个时间步长内，都有一个联合动作 $\mathbf{a} = \langle a^1, \dots, a^n \rangle$ 发生。通过转换函数 T 描述联合动作如何影响环境，也具体说明了 $P(s'|s, \mathbf{a})$ 。在一个 Dec-POMDP 中，学习器只知道它们自己的动作；它们不观察彼此的动作，和联合动作集相似， $\mathcal{O} = \times_{i \in \mathcal{D}} \mathcal{O}^i$ 是一个联合观测集， \mathcal{O}^i 是指学习器 i 可能的观测集，每一个时间步长内，环境发出一个联合观测集 $\mathbf{o} = \langle o^1, \dots, o^n \rangle$ ，根据这个观测集，每一个学习器 i 也只观测到自身的观测值 o^i ，观测函数 o 阐明了联合观测集的可能性 $P(o|\mathbf{a}, s')$ ，图 15.2 进一步说明了 Dec-POMDP 模型的动态过程。

在执行期间，假定学习器仅在个别观测的基础上动作而且没有额外的通信，但这并不意味着 Dec-POMDP 不能建立和通信相关的模型。例如，如果一个学习器有一个“标记黑板”的动作并且其他学习器观测“被标记的黑板”，这些学习器通过环境的状态有一个通信机制。然而，我们说 Dec-POMDP 能通过动作、状态以及观测更隐式地建立通信模型，而不是让这种通信明确化。这意味着在一个 Dec-POMDP 中，通信并没有特别的含义，15.4.4 节进一步

⊖ 在某些情况下，假设学习器已经得到联合策略，这使我们能够根据本地广播的观测值计算联合信任值（见 15.5.4 节）。

⊖ 另外，每个学习器并行运行相同的规划算法。

阐述了 Dec-POMDP 中的通信。

本章的重点是有限范围内的规划，(无折扣的) 预期累积奖励是其常用的优化准则。规划问题因而意味着要找到一个策略元组，称为联合策略，它使预期累积奖励最大化。

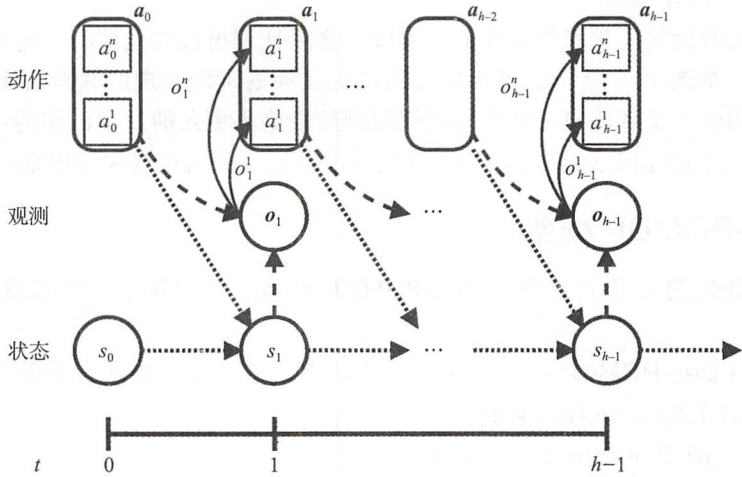


图 15.2 Dec POMDP 动态流程更详细的说明。在每个阶段，环境都处在一个特定的状态，这种状态根据观测模型发出一个联合观测函数集（虚线箭头），每个学习器从中得到自身的观测函数（由实心箭头表示），然后每个学习器选择一个动作，一起形成联合动作，根据一个转换模型，这些联合动作导致一次状态转换发生（点线箭头）

我们考虑分散的老虎 (Dec-Tiger) 问题 (由 [Nair et al ,2003 c] 介绍的常用的 Dec-POMDP 基准) 作为一个例子。它关注在一个有两扇门的走廊里的两个学习器。一扇门后面是一只老虎，另一扇门后面是一件宝物。因此有两个状态：老虎在左门 (s_l) 或右门 (s_r) 后面。每一个学习器在它们的处理过程中都有 3 个动作：开左门 (a_{OL})、开右门 (a_{OR}) 和听 (a_{Li})。它们不能观测到彼此的动作。实际上，它们只能收到两种观测值：可以听左边或者右边门后面的老虎发出声音。

当 $T = 0$ 时状态是 s_l 或 s_r 的概率都是 0.5。只要没有学习器开门，状态就不会改变。当一扇门打开时，状态将重置为概率为 0.5 的 s_l 或 s_r ，每个学习器的观测概率都是相同且互相独立的。例如，当前状态是 s_l 且两个执行操作都是 a_{Li} ，每个学习器都有 85% 的机会观测 o_{HL} ，则听到左边或者右边门后面是老虎的概率都是 $0.85 \times 0.85 = 0.72$ 。当其中一个学习器打开门后是宝物的这一扇门，它们得到奖励 (+9)，他们若打开门后是老虎的那扇门，则受到惩罚 (-101)，当共同打开错误的门，惩罚较少 (-50)，当共同打开正确的门，就会获得更多的奖励 (+20)。所有转换过程、观测、奖励模型由 [Nair et al , 2003 c] 列出。

请注意，当一个或两个学习器同时开错门，它们则受到老虎攻击并受到惩罚。然而学习器既没有观测到这种攻击也没观测到惩罚 (记住，唯一可能的观测是开左门 o_{HL} 和开右门 o_{HR})，情节继续发展。直观地说，分散老虎问题的最优联合策略应该指定学习器一直在听直到它们足够确定打开哪一扇门。同时，策略应该尽可能“协调”，使联合动作的可能性达到最大。

15.3 历史状态与策略

在 MDP 中，学习器使用从状态映射到动作的策略。在选择动作时，由于马尔可夫属

性, 它可以忽略历史。在 POMDP 中学习器可以不再观察状态, 但它要计算一个总结历史的信任值 b , 这个 b 也是一个马尔可夫信号。然而, 在 Dec-POMDP 执行期间, 每个学习器只能得到其个别动作值和观测值, 且现在还没有一个已知的方法来总结个别历史。用和 POMDP 相同的方式维护并更新一个个别信任值是不可能的, 因为转换和观测函数在联合动作和观测方面被阐明^①。

这意味着, 在 Dec-POMDP 中学习器在执行过程中没有获得马尔可夫信号。这样的后果是, Dec-POMDP 问题的规划将涉及搜索从全面的个人历史映射到动作的个别 Dec-POMDP 策略的元组空间。我们稍后会明白, 这意味着解决 Dec-POMDP 问题甚至比解决 POMDP 问题还困难。

15.3.1 历史状态

首先, 我们定义观测历史、动作历史以及动作观测历史。

定义 15.2 (动作观测历史) 学习器 i 的动作观测历史 (AOH) $\bar{\theta}^i$ 是学习器 i 采取的动作序列和接收的观测序列, 在一个特定的时间步长 t 中, $\bar{\theta}_t^i = (a_0^i, o_1^i, \dots, o_{t-1}^i, o_t^i)$ 。

联合动作观测历史 $\bar{\theta}_t = \langle \bar{\theta}_t^1, \dots, \bar{\theta}_t^n \rangle$ 详细阐明了所有学习器的 AOH, 学习器 i 在时间 t 时可能的 AOH 集是 $\bar{\Theta}_t^i$, 学习器 i 所有状态可能的 AOH 集是 $\bar{\Theta}^i$, $\bar{\theta}^i$ 表示这个集合中的一个 AOH^②。最终所有可能的联合 AOH 集合 $\bar{\Theta}$ 表示为 $\bar{\Theta}$, 在 $t=0$ 时刻, 联合 AOH 为空, 即 $\bar{\theta}_0 = ()$ 。

定义 15.3 (观察历史) 将学习器的观察历史 (OH) \bar{o}^i 定义为一个学习器接收的观测序列, 在一个特定的时间步长 t 内, $\bar{o}_t^i = (o_1^i, \dots, o_t^i)$ 。

联合观测历史是所有学习器的 OH: $\bar{o}_t = \langle \bar{o}_t^1, \dots, \bar{o}_t^n \rangle$ 。学习器 i 在时刻 t 的观察历史集表示为 \bar{O}_t^i , 和动作观测历史的标记方法相似, 我们同样用 $\bar{o}^i \in \bar{O}^i$ 和 $\bar{o} \in \bar{O}$ 表示。

定义 15.4 (动作历史) 学习器 i 的动作历史 (AH) \bar{a}^i 是一个学习器发生过的动作序列 $\bar{a}_t^i = (\bar{a}_0^i, \bar{a}_1^i, \dots, \bar{a}_{t-1}^i)$ 。

联合动作历史和集合的符号与那些观察历史的符号相似, 最后注意, 一个联合 AOH 由一个联合动作和一个联合观察历史构成: $\bar{\theta}_t = \langle \bar{o}_t, \bar{a}_t \rangle$ 。

15.3.2 策略

学习器 i 的策略 π^i 从历史映射到动作。在一般情况下, 这些历史是动作观测历史 (AOH), 因为它们包含一个学习器的所有信息, 从问题的范围看, AOH 呈指数级增长: 在时间步长 t 内, 学习器 i 有 $(|A^i| \times |O^i|)^t$ 种可能的 AOH。策略 π^i 给每一个历史状态指定动作, 所以, 可能的策略 π^i 是成倍指数幂级规模的。

通过为具有上面考虑过的形式的策略指定相同的动作, 以此减少需要考虑的策略的数量是可能的。通过图 15.3a 说明: 在一个确定的策略下, 只有动作观测历史的一个可能的子集

① [Nair et al, 2003c; Hansen et al, 2004; Oliehoek et al, 2009; Zettlemoyer et al, 2009] 研究过类 Dec-POMDP 模型的不同形式的信任值, 它们不仅仅由状态决定, 还指定了历史状态、策略、类型以及其他学习器的信任值的概率。关键是从单个学习器的角度看, 仅知道状态的概率分布是不够的, 还需要预测其他学习器将采取的行为。

② 在一个特定的去中心化的部分可观察马尔可夫决策过程问题中, 可能出现的情况是并不能真正达到所有历史状态, 因为这些可能性由转换和观测模型决定。

能实现。相对于不能实现 AOH 的仅有一点区别的策略，表现出相同的动作。结果，为了指定一个确定性的策略，仅有观测历史是不够的：当一个学习器确定性地选择一个动作时，它可以仅从观测历史中指出要采取的动作。这意味着一个确定性的策略能用一棵策略树方便地表示，如 15.3b 所示。

476 定义 15.5 学习器 i 的一个确定性的策略 π^i 是从观察历史到动作的映射 $\pi^i: \bar{O}^i \rightarrow A^i$ 。

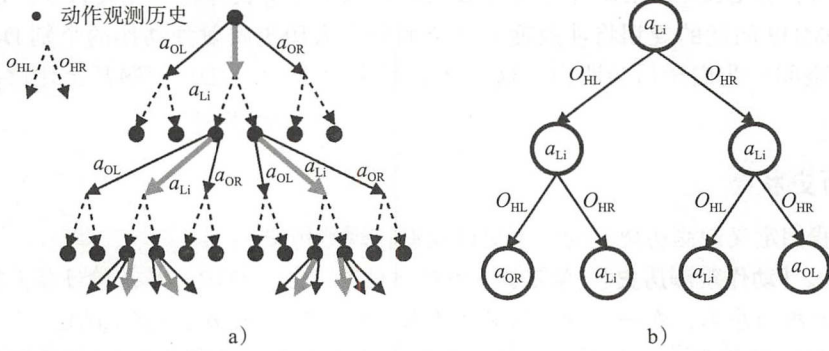


图 15.3 确定性的策略可以表示为一棵树。a) Dec-Tiger 问题中一个学习器的一棵动作观测历史树 $\bar{\theta}^i$ 。确定性策略 π^i 被突出显示。我们清楚地看到， π^i 仅到达了历史 $\bar{\theta}^i$ 的子集（没到达的 $\bar{\theta}^i$ 没有被进一步扩展）。b) 相同的策略也能用简化策略树表示。在 $h=3$ 的 Dec-Tiger 问题中，两个学习器都执行这种策略时，联合策略是最优的

对于确定性的策略， $\pi^i(\bar{\theta}^i)$ 表示它为包含在 $\bar{\theta}^i$ 中的观测历史指定的操作。例如，让 $\bar{\theta}^i = \langle \bar{o}^i, \bar{a}^i \rangle$ ，所以， $\pi^i(\bar{\theta}^i) \triangleq \pi^i(\bar{\theta}^i)$ ，我们用 $\pi = \langle \pi^1, \dots, \pi^n \rangle$ 来表示联合策略。我们说一个确定性的联合策略是从联合观测历史到联合动作的诱导映射 $\pi: \bar{O} \rightarrow A$ 。映射被构成联合策略的个别策略 π^i 诱导。但请注意，只有可能映射 $f: \bar{O} \rightarrow A$ 的一个子集相当于有效的联合策略：若 f 没有为每一个学习器 i 的每一个 \bar{O}^i 指定相同的个别动作，它也不可能以一个分散的方式执行 f 。也就是说，这种策略是集中式的：它描述了一个学习器应该在联合历史的基础上选择动作。然而，在执行过程中它也只能得到个别观测历史，而不是联合观测历史。

学习器还可以执行随机策略，但我们把注意力局限于确定性策略而不牺牲最优性，因为一个有限域的 Dec-POMDP 至少具有一个最优纯联合策略 [Oliehoek et al, 2008b]。

15.3.3 策略的结构

策略为 Dec-POMDP 的所有状态指定动作。在一个策略中代表临时结构的共同方式是为每个状态划分指定策略的决策规则 δ^i 。一个个别策略可表示为一个决策规则序列 $\pi^i = (\delta_0^i, \dots, \delta_{h-1}^i)$ 。如果是一个确定性策略，阶段 t 决策规则的形式就是一个 t 长度的观测历史到动作的映射 $\delta_t^i: \bar{O}_t^i \rightarrow A^i$ 。在更一般的情况下它的范围是一个 AOH 集 $\delta_t^i: \bar{\Theta}_t^i \rightarrow A^i$ ，联合决策规则 $\delta_t = \langle \delta_t^1, \dots, \delta_t^n \rangle$ 为每一个学习器指明一个决策规则。

我们也会考虑部分随时间指定的策略。通常情况下， $\varphi_t = (\delta_0, \dots, \delta_{t-1})$ 表示阶段 t 的过去联合策略，它指明了阶段 0 到 $t-1$ 的部分联合策略，通过为阶段 t 追加一个联合决策规则，我们能得到这样一个过去联合策略。

定义 15.6 (策略相关性) 我们用

$$\varphi_{t+1} = (\delta_0, \dots, \delta_{t-1}, \delta_t) = \langle \varphi_t, \delta_t \rangle \quad (15.1)$$

表示策略相关性。

学习器 i 的未来策略 ψ^i 指明了阶段 t 的所有未来动作, 即 $\psi^i = (\delta_{t+1}^i, \dots, \delta_{h-1}^i)$, 未来联合策略集表示为 $\Psi = (\delta_{t+1}, \dots, \delta_{h-1})$, 策略 π^i 的结构被表示为

$$\pi^i = (\underbrace{\delta_0^i, \delta_1^i, \dots, \delta_{t-1}^i}_{\varphi_t^i}, \underbrace{\delta_t^i, \delta_{t+1}^i, \dots, \delta_{h-1}^i}_{\psi_t^i}) \quad (15.2)$$

与联合策略相似。

由于策略能表示为树 (见图 15.3), 一种不同的分析方法就是通过考虑子树。阶段 t 的剩余时间 τ 定义为 $\tau = h - t, q_\tau^i = k$ 表示学习器 i 的一个 k 步剩余子树策略, 即 $q_\tau^i = k$ 是一个和 k 规模问题的全策略有相同形式的策略树, 在原始 h 规模问题中 $q_\tau^i = k$ 是一个在阶段 $t = h - k$ 时刻开始的候选执行操作, 学习器 i 的 k 步剩余子树策略集表示为 $Q_\tau^i = k$, 联合子树策略 $q_{\tau=k} \in Q_{\tau=k}$ 为每一个学习器指明一个子树策略。

图 15.4 展示了一个策略中 $h=3$ 时虚构 Dec-POMDP 的不同结构, 它用带点的椭圆表示, 同时它也展现了一个过去策略 φ_2^i 并解释了策略相关 $\langle \varphi_2^i \circ \delta_2^i \rangle = \pi^i$ 如何形成全策略。全策略也和一个 3 步剩余子树策略 $q_\tau^i = 3$ 相关, 两个子树策略用虚线椭圆形表示。

定义 15.7 (策略假设) 假设有 l ($l < k$) 个观测值的 k 步剩余子树策略 $q_\tau = k$ 损耗了导出一个联合子树策略 $q_\tau = k$ 的一部分, 这个联合子树策略是 $q_\tau = k$ 的一棵子树, 特别地, 单个联合观察 o 产生的消耗 \downarrow 写为

$$q_{\tau=k-1} = q_{\tau=k} \downarrow_o \quad (15.3)$$

例如, 在图 15.4 中, $q_{\tau=1}^i = q_{\tau=2}^i \downarrow_{\delta^o}$

478

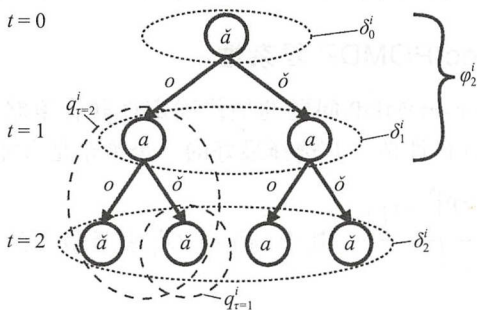


图 15.4 带有动作 $\{a, \bar{a}\}$ 和观测 $\{o, \bar{o}\}$ 的某学习器的策略结构, 一个策略 π^i 能被划分为决策规则 δ^i 或子树策略 q^i 。

15.3.4 联合策略的质量

联合策略期望积累的奖励值是不同的, 这将作为衡量它们质量的一个标准, 正式地, 我们考虑一个联合策略的期望累积奖励, 同时也可看作该策略的价值。

定义 15.8 联合策略 π 的价值 $V(\pi)$ 定义为 $V(\pi) \triangleq E \left[\sum_{t=0}^{h-1} R(s_t, a_t) | I, \pi \right]$, 这里求的是状态和观测值的期望。

期望值能用一个递归式计算, 在上一个阶段 $t = h - 1$ 下, 价值只是由当前奖励给出

$$V^\pi(s_{h-1}, \bar{o}_{h-1}) = R(s_{h-1}, \pi(\bar{o}_{h-1}))$$

对于所有其他状态, 其价值由

$$V^\pi(s_t, \bar{o}_t) = R(s_t, \pi(\bar{o}_t)) + \sum_{s_{t+1} \in S} \sum_{o \in O} P(s_{t+1}, o | s_t, \pi(\bar{o}_t)) V^\pi(s_{t+1}, \bar{o}_{t+1}) \quad (15.4)$$

给出。这里的概率只是转换和观测概率的乘积 $P(s', o | s, a) = P(o | a, s') P(s' | s, a)$ 。实质上，确定的联合策略能将 Dec-POMDP 转换为状态 (s_t, \bar{o}_t) 下的马尔可夫链。通过动态规划评估这个方程将得出所有 (s_0, \bar{o}_0) 对的值。根据初始状态分布 I ， $V(\pi)$ 的值通过给这些对的加权给出，
 [479] 注意，给定确定的联合策略 π ，历史 \bar{o}_t 实际上产生了一个联合子树策略，照这样，就子树策略而言，改写公式 (15.4) 是可能的。在最后 k 个阶段执行 $q_{t=k}$ ，在 $t = h-k$ 时从状态 s 开始将实现

$$V(s_t, q_{t=k}) = R(s_t, a_t) + \sum_{s_{t+1} \in S} \sum_{o \in O} P(s_{t+1}, o | s_t, a_t) V(s_{t+1}, q_{t=k} \downarrow_o) \quad (15.5)$$

其中 a_t 是由 $q_{t=k}$ 的根指定的联合动作。

最后，就像上述方程展现的，状态的概率和历史在很多计算中都是重要的，下面的方程递归地指定了状态 I 和 (潜在可能的) 过去联合策略 φ_t 下联合动作观测历史 AOH 的概率：

$$P(s_t, \bar{\theta}_t | I, \varphi_t) = \sum_{s_{t-1} \in S} \sum_{a_{t-1} \in A} P(s_t, o_t | s_{t-1}, a_{t-1}) P(a_{t-1} | \bar{\theta}_{t-1}, \varphi_t) P(s_{t-1}, \bar{\theta}_{t-1} | I, \varphi_t) \quad (15.6)$$

15.4 有限域的 Dec-POMDP 的解决方案

本节提出了寻找有限区间的 Dec-POMDP 问题确切的近似方法的概述。对于无限范围的 Dec-POMDP 问题，情况就明显不同了，在 15.5 节将给出一些说明。

15.4.1 穷举搜索和 Dec-POMDP 复杂性

因为存在有限区间 Dec-POMDP 问题的最优确定性联合策略，所以枚举所有的联合策略、像 15.3.4 节所述对其进行评估、并选择最好的一个策略是可能的。然而，计算这类联合

策略的数量级是 $O\left(|A^\dagger| \frac{n(|O^\dagger| - 1)}{|O^\dagger| - 1}\right)$ ，其中 $|A^\dagger|$ 和 $|O^\dagger|$ 表示最大的个人动作和观测集，估计

计算每一个联合策略的时间复杂度都是 $O(|S| \times |O^\dagger|^n)$ ，很清楚这种方法只适合于 n 非常小的问题，这种分析意味着问题是多么困难，这种直觉由 [Bernstein et al, 2002] 发表的复杂结果支撑。

定理 15.1 (Dec-POMDP 复杂性) 寻找有限区间 Dec-POMDP ($n \geq 2$) 问题的最优解是 NEXP 完全的。

NEXP 指的是解决该问题所花费的不确定的指数级时间的等级。不确定性意味着，类似
 [480] 于 NP 问题，它需要以一种不确定的方式产生关于一个解决方案的猜想。计算时间服从指数分布意味着验证该猜想是否是一种解决方案的时间复杂度为指数级。在实践中这意味着，在最坏的情况下 (假定 $\text{NEXP} \neq \text{EXP}$) 解决 Dec POMDP 问题需要成倍的指数幂时间。此外，Dec-POMDP 问题没有有效的近似解：[Rabinovich et al, 2003] 表明，找到一种 ε 近似解甚至也是 NEXP 完全的。

15.4.2 交替最大化

基于搜索的策略的联合均衡 (JESP) [Nair et al, 2003c] 是一种保证能够找到一个局部

最优的联合策略方法，更具体地说，是纳什均衡：对于每个学习器 i 的策略元组，其策略 π^i 是其他的学习器采用的策略 π^{-i} 的最优反馈。这依赖于一个称为交替最大化的过程。这是一个为使联合奖励最大化而计算策略 π^i 的过程，在该过程中也能确定其他学习器的策略。其次，另一个学习器通过找到它的最优反馈来使联合奖励最大化。这个过程一直重复直到联合策略达到纳什均衡，即达到局部最优。这一过程也称为爬山法或坐标上升法。请注意，局部最优的结果可以是任意差的，比如，在 Dec-Tiger 问题中，如果学习器 1 马上开左门 (a_{OL})，那么学习器 2 最优反馈结果就是选择 a_{OL} ，为了减少这样一个差的局部最优值所带来的影响，JESP 可以使用随机数重启。

JESP 使用动态规划方法来计算一个选定的学习器 i 的最优反馈策略，本质上，确定 π^i 允许重新阐述一个扩充的 POMDP。在这个强化 POMDP 中，状态 $\tilde{s} = \langle s, \bar{o}^{-i} \rangle$ 构成一个候选状态 s 以及其他学习器的观测历史 \bar{o}^{-i} ，给定其他学习器的确定性策略 π^{-i} ，扩充状态 \tilde{s} 就是马尔可夫的，且所有的转换和观测概率都能由 π^{-i} 和原始 Dec-POMDP 问题的转换和观测模型推导而来。

15.4.3 Dec-POMDP 的最优价值函数

本节介绍了一种更符合单学习器 MDP 问题和 POMDP 问题的方法：我们可以确定用于派生最优策略的最优价值函数 Q^* ，即使计算 Q^* 很困难，但它的内部机制却很有价值。尤其，它与两个解决 Dec-POMDP 问题的主要方法都有一个明确的关系，前向推导和后向推导的方法将在接下来的小节中说明。

481

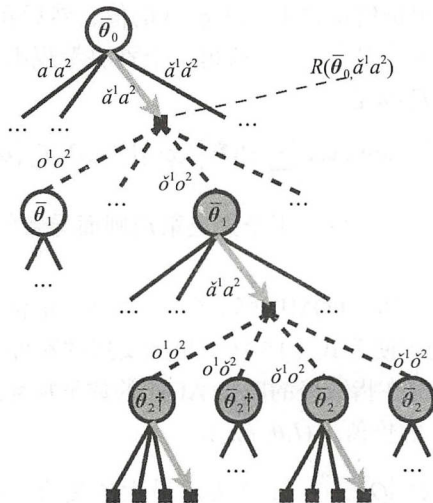


图 15.5 动作集为 $\{\{a^1, \bar{a}^1\}, \{a^2, \bar{a}^2\}\}$ 和样本集为 $\{\{o^1, \bar{o}^1\}, \{o^2, \bar{o}^2\}\}$ 的虚构双学习器联合 AOH 树 $\bar{\theta}$ ，给出 I ，AOH 通过状态产生一个联合信任值 $b(s)$ ，实线代表联合动作，虚线代表联合观测值。由于树的大小，它只显示了一部分，突出强调了联合动作所代表的一种联合策略。给定一个节点的联合子树策略（在下面的子树中做的动作选择），其值由公式 (15.8) 给出。然而，树的不同部分的动作选择不是独立的：例如，两个标记有 \dagger 的节点有相同的 $\bar{\theta}^1$ ，因此应为学习器 1 指定相同的子树策略

15.4.3.1 选择子树策略

让我们通过图 15.5 开始考虑，它是一棵表示联合 AOH 的树，对于一个特定的联

合 AOH (图 15.5 中的节点), 我们能尝试确定哪一个联合子树策略 $q_{\tau=k}$ 是最优的, 回想 $V(s_t, q_{\tau=k})$, 从 s_t 开始 $q_{\tau=k}$ 的值由公式 (15.5) 指明, 而且, 让 $b(s) \triangleq P(s|I, \bar{\theta}_t)$ 成为和 $\bar{\theta}_t$ 相关的联合信任值, 这个联合信任值能像 PODMP 信任值更新那样用贝叶斯规则以相同的方式计算的, 鉴于初始信任值 I 和联合 AOH $\bar{\theta}_t$, 我们能通过

$$V(I, \bar{\theta}_t, q_{\tau=k}) = \sum_{s \in S} b(s) V(s, q_{\tau=k}) \quad (15.7)$$

为每一个学习器子树策略 $q_{\tau=k}$ 计算一个能被当前节点使用的值, 现在, 可以递归地重写公式 (15.7):

$$V(I, \bar{\theta}_t, q_{\tau=k}) = R(\bar{\theta}_t, a_t) + \sum_o P(o|b, a) V(I, \bar{\theta}_{t+1}, q_{\tau=k} \downarrow_o) \quad (15.8)$$

期望的即时奖励由

$$R(\bar{\theta}_t, a_t) = \sum_{s_t \in S} b(s_t) R(s_t, a) \quad (15.9)$$

给出。

因此我们希望动态规划方法是可行的 $\bar{\theta}_t$, 每个可以选择最大的 $q_{\tau=k}$ 。不幸的是, 由于 Dec-POMDP 去中心化的性质, 在整棵树上运行这种程序是不可能的: 独立地选择最大的联合子树策略 $q_{\tau=k}$ 是不可能的, 因为这可能导致集中的联合策略。

结果, 即使公式 (15.8) 可用于计算 $(\bar{\theta}_t, q_{\tau=k})$ 对的值, 它对优化联合策略也没有直接的帮助, 因为我们不能独立地推理出联合 AOH 树的一部分, 相反, 应该决定通过假设一个过去联合策略 φ_t 考虑相同时间内一个阶段 t 所有的 $\bar{\theta}_t$ 来决定选择哪一棵子树策略, 我们假设已经为所有的 $\bar{\theta}_t$ 和 $q_{\tau=k}$ 计算出价值函数 $V(I, \bar{\theta}_t, q_{\tau=k})$ 的值, 然后我们就可以为阶段 t 计算一种特殊形式的联合决策规则 $\Gamma_t = \langle \Gamma_t^i \rangle_{i \in \mathcal{D}}$, 这里, 个别决策规则将个别历史映射到个别子树策略 $\Gamma_t^i: \bar{\theta}_t^i \rightarrow Q_{\tau=k}^i$, 最优的 Γ_t 满足:

$$\Gamma_t^* = \arg \max_{\Gamma_t} \sum_{\bar{\theta}_t \in \bar{\Theta}_t} P(\bar{\theta}_t | I, \varphi_t) V(I, \bar{\theta}_t, \Gamma_t(\bar{\theta}_t)) \quad (15.10)$$

其中 $\Gamma_t(\bar{\theta}_t) = \langle \Gamma_t^i(\bar{\theta}_t^i) \rangle_{i \in \mathcal{D}}$ 表示由采用个别决策规则而导致的联合子树策略 $q_{\tau=k}$, 概率是公式 (15.6) 的边缘概率。

该方程清楚地表明了一个 Dec-POMDP 问题在 t 时刻的最优联合策略取决于 φ_t , 联合策略追踪阶段 t , 此外, 还有用于使公式 (15.10) 更加实用的额外复杂条件:

1) 将学习器和 t 的数量都在指数级的联合 AOH 的数量加起来。

2) 假设计算所有 $\bar{\theta}_t, q_{\tau=k}$ 的价值 $V(I, \bar{\theta}_t, q_{\tau=k})$ 。

3) 估算 Γ_t 的数量级是 $O(|Q_{\tau=k}^\dagger|^{|\bar{\Theta}_t^\dagger|})$, \dagger 表示最大的集合, $|Q_{\tau=k}^\dagger|$ 是双重指数级的, $|\bar{\Theta}_t^\dagger|$

是指数级的, 因此 Γ_t 的数量级在 $h = t+k$ 时是双重指数级的。

请注意, 通过限制我们对确定性 φ_t 的关注, 重新定义公式 (15.10) 是可能的, 将其作为一个观测历史 OH 的总和, 而不是动作观测历史 AOH (这涉及将 φ_t 作为参数的可适应的 V) 的总和。然而, 对于这种重新规定, 在相同的复杂条件下也成立。

15.4.3.2 选取最优决策规则

本节将焦点转换到常规决策规则 δ^i , 像 15.3.3 节中介绍的那样——从观测历史 OH (或 AOH) 映射到动作。与由 δ_t^i 指定一样, 我们指定一个价值函数, 将所采取动作的期望值量化, 并持续地优化它, 即我们通过决策规则的最优值取代公式 (15.10) 中子树的值, 有限

域的 Dec-POMDP 问题的最优价值函数定义如下。

定理 15.2 (最优 Q^* 值) 最优 Q 值函数 $Q^*(I, \varphi_t, \bar{\theta}_t, \delta_t)$ 是一个初始分布状态和过去联合策略、AOH 和决策规则的函数, 由于在上一个阶段,

$$Q^*(I, \varphi_{h-1}, \bar{\theta}_{h-1}, \delta_{h-1}) = R(\bar{\theta}_{h-1}, \delta_{h-1}(\bar{\theta}_{h-1})) \quad (15.11)$$

就像公式 (15.9) 定义的那样, 对于所有的 $0 \leq t < h-1$,

$$Q^*(I, \varphi_t, \bar{\theta}_t, \delta_t) = R(\bar{\theta}_t, \delta_t(\bar{\theta}_t)) + \sum P(o|\bar{\theta}_t, \delta_t(\bar{\theta}_t)) Q^*(I, \varphi_{t+1}, \bar{\theta}_{t+1}, \delta_{t+1}^*) \quad (15.12)$$

其中, $\varphi_{t+1} = \langle \varphi_t \circ \delta_t \rangle$, $\bar{\theta}_{t+1} = (\bar{\theta}_t, \delta_t(\bar{\theta}_t), o)$,

$$\delta_{t+1}^* = \arg \max_{\delta_{t+1}} \sum_{\bar{\theta}_{t+1} \in \bar{\Theta}_{t+1}} P(\bar{\theta}_{t+1} | I, \varphi_{t+1}) Q^*(I, \varphi_{t+1}, \bar{\theta}_{t+1}, \delta_{t+1}) \quad (15.13)$$

证明 由于公式 (15.11), 为上一个阶段应用了公式 (15.13) 将使期望的奖励值最大化, 因此是最优的, 公式 (15.12) 给先前的状态产生最优值, 所有阶段的最优化归纳如下。

注意, 为了计算下一个状态的最优联合决策 δ_{t+1}^* , φ_t 是必须的, 因为公式 (15.13) 需要求出 $\varphi_{t+1|0}$

上述方程组成一个动态规划, 当假设只有确定的过去联合策略 φ 能够使用, 动态规划则能从结果 ($t = h-1$) 到开始 ($t = 0$) 进行评估, 图 15.6 说明了 Q^* 的计算过程, 当到达阶段 0 的时候, 过去联合策略为空 $\varphi_0 = ()$, 联合决策规则只是联合动作, 因此选择

$\delta_0^* = \arg \max_{\delta_0} Q^*(I, \varphi_0, \bar{\theta}_0, \delta_0) = \arg \max_a Q^*(I, (), (), a)$ 是可能的。

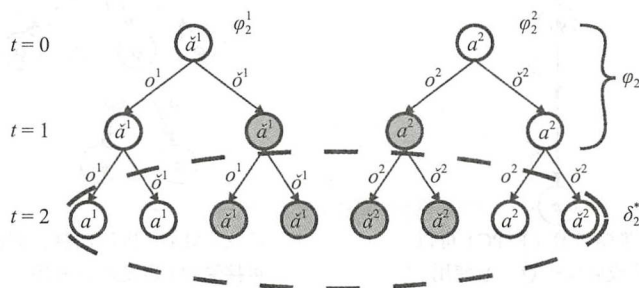


图 15.6 计算 Q^* 。虚线椭圆表示阶段 $t=2$ 时规则 δ_2^* 的最优决策, 假设 $\varphi_2 = \langle \varphi_1, \delta_1 \rangle$ 遵循前两个状态, $Q^*(I, \bar{\theta}_1, \varphi_1, \delta_1)$ 通过传播下一个状态的相关 Q^* 值来计算, 例如, 为突出联合历史 $\bar{\theta}_1 = \langle (a^1, o^1), (a^2, o^2) \rangle$, φ_2 下的 Q^* 值通过传递四个接替的联合历史, 下的值来计算, 如公式 (15.12) 所示

给定 $\varphi_1 = \delta_0^*$ 时, 我们能利用公式 (15.13) 确定 δ_1^* 等^①。这个过程, 我们用 Q^* 指向前掠形策略计算 (FSPC), FSPC 的原则是从迄今为止发现的过去联合策略中选择一个新的决策规则并在图 15.7a 中给出其说明。

不幸的是, 计算 Q^* 本身是很难的, 因为这意味着要为所有的过去联合策略评估定理 15.2 所述的动态规划。尤其是, 需要为所有的 $(\varphi_{h-1}, \delta_{h-1})$ 评估公式 (15.11), 且这些对和所有的联合策略直接相关: $\pi = \langle \varphi_{h-1} \circ \delta_{h-1} \rangle$, 因此, 评估这个 Dec-POMDP 问题的时间为 h 的成倍指数级, 这意味着 Q^* 的实际值是受限的。

当遇到 MDP 和 POMDP 问题时, 被识别的 Q^* 和 Q 值函数的形式有很大不同, 我们仍然

① 请注意: 已求出而且存储了公式 (15.13) 中的最大值。

使用符号 Q , 因为 δ_t 能看成在计算过程元水平的一个动作, 在这个过程中 (I, φ_t) 能被翻译成状态而且我们能它们的通用含义来定义 V 和 Q , 尤其可能写成

$$V^*(I, \varphi_t) = \max_{\delta_t} Q^*(I, \varphi_t, \delta_t) \quad (15.14)$$

其中 Q^* 定义为

$$Q^*(I, \varphi_t, \delta_t) = \sum_{\bar{\theta}_t} P(\bar{\theta}_t | I, \varphi_t) Q^*(I, \varphi_t, \bar{\theta}, \delta_t)$$

通过利用公式 (15.12) 扩充 Q^* 的定义, 我们能验证它是否确实能通过首次采取动作 δ_t 加上之后持续最理想的积累奖励对期望的立即奖励有一个合理说明 [Oliehoek, 2010]。

15.4.4 前推法: 启发式搜索

之前的小节中, 曾计算了 Q^* , 说明通过演示前掠策略的计算提取 π^* 是可能的: 对连续阶段 $t=0, 1, \dots, h-1$ 重复运用公式 (15.13)。当处理阶段 t 时, 阶段 $0 \dots t-1$ 已经被处理完, 因此一个过去联合策略 $\varphi_t = (\delta_0, \dots, \delta_{t-1})$ 是可以实现的, 而且其概率 $P(\bar{\theta}_t | I, \varphi_t)$ 已被定义, 不幸的是, 计算 Q^* 本身是困难的, 克服该问题的一个思路是用一个容易计算的近似值 \hat{Q} , 我们把这个称作 Dec-POMDP 的前推式方法。

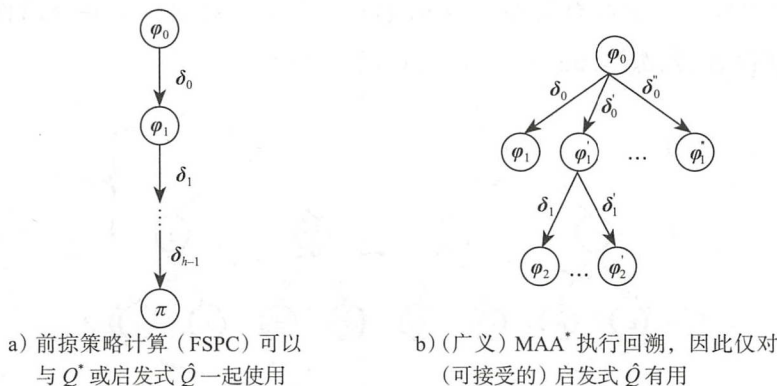


图 15.7 Dec-POMDP 的前推式方法

15.4.4.1 Dec-POMDP 作为系列贝叶斯博弈

一个直接方法就是尝试运用一个使用启发式 Q 值函数 \hat{Q} 的前推式策略计算, 这本质上就是 [Emery-Montemerlo et al, 2004] 所介绍的方法, 它利用一系列协作贝叶斯博弈模型 (CBG) 代表一个去中心化的部分可观察马尔可夫决策过程。每一个阶段 t , 都有一个近似收益函数 $\hat{Q}(\bar{\theta}_t, a)$ 。贝叶斯博弈模型 [Osborne and Rubinstein, 1994] 是有私有信息的学习器策略性博弈的一个扩展。CBG 是一个有相同收益的贝叶斯博弈。通过解决连续状态的 CBG 问题, 能找到一个近似解, 这就是前推式策略计算方法 (利用 \hat{Q})。

在一个 Dec-POMDP 问题中, 在某个阶段 t 做决策的关键难题在于学习器缺少一个以它们的动作为条件的共同信号, 相反, 它们必须在个人历史的基础上动作, 给定 I 和 φ_t , 这种情况能建立一个 CBG 模型, 这样一个 CBG $B(I, \varphi_t)$ 的组成是:

- 学习器集
- 联合动作 A
- 联合 AOH 集 $\bar{\theta}_t$

- 分布的概率 $P(\bar{\theta}_i|I, \varphi_i)$
- 收益函数 $\hat{Q}(\bar{\theta}_i, \mathbf{a})$

486

在 CBG 问题中, 学习器利用从个人 AOH 映射到动作的策略, 即, 学习器 i 的 CBG 策略与 Dec-POMDP 问题的决策规则 δ_i^t 相关, CBG 的解决方案是考虑 \hat{Q} 使期望收益最大化的联合决策规则: δ_i

$$\hat{\delta}_i^* = \arg \max_{\delta_i} \sum_{\bar{\theta}_i \in \Theta_i} P(\bar{\theta}_i|I, \varphi_i) \hat{Q}(\bar{\theta}_i, \delta_i(\bar{\theta}_i)) \quad (15.15)$$

如果 φ_i 确定, 则 $\bar{\theta}_i = \langle \bar{\mathbf{a}}_i, \bar{\mathbf{o}}_i \rangle$ 的概率对于 $\bar{\mathbf{a}}_i$ 都是非零的, 这意味着我们可以把注意力限定在 OH 以及从 OH 映射到动作的决策规则上。

15.4.4.2 启发式 Q 值函数

通过给出 I, φ_i 和 \hat{Q} , 一个状态的 CBG 模型就能完全确定, 但怎样确定 \hat{Q} 并不明确, 这里我们将讨论这个问题。

注意, 由于上一个阶段 $t = h-1, \hat{\delta}_i^*$ 与公式 (15.13) 选择的最优决策规则有着相近的联系^①, 如果由于上一个阶段, 启发式函数指明了当前奖励 $\hat{Q}(\bar{\theta}_i, \mathbf{a}) = R(\bar{\theta}_i, \mathbf{a})$, 则两个都要选择相同的动作, 即在 $\hat{\delta}_i^* = \delta_i^*$ 的条件下。

然而对于所有的阶段指明这样一个强大的相关性是不可能的, 注意到通过 CBG 模型实现的 FSPC 并不是最优子解: 也能为任意 π 计算一个形式为 $Q^\pi(\bar{\theta}_i, \mathbf{a})$ 的价值函数, 为一个 π^* 域 Q^{π^*} 实现这个, 以及为 CBG 模型使用后者作为收益函数时, FSPC 是准确的 [Oliehoek et al, 2008b]^②。

然而, 这种洞察力的实际价值是有限的, 因为它要求开始要知道一个最优策略, 实际上, 计算潜在 MDP 问题的价值函数 $Q_M(s, \mathbf{a})$ 是可能的: MDP 和 Dec-POMDP 有相同的转换和奖励函数 [Emery-Montemerlo et al, 2004; Szer et al, 2005], 这能用于计算 CBG 模型的收益函数 $\hat{Q}(\bar{\theta}_i, \mathbf{a}) = \sum_s \mathbf{b}(s) Q_M(s, \mathbf{a})$, 称作 Q_{MDP} 。相似地, 使用潜在 POMDP 问题 (Q_{POMDP}) 的价值函数也是可能的 [Roth et al, 2005b; Szeretal, 2005], 或者使用一步延迟通信问题的价值函数 (Q_{BG}) 也是可能的 [Oliehoek and Vlassis, 2007]。

FSPC 的一个问题是公式 (15.15) 仍然最大化从历史到动作的映射 δ_i , 这样的 δ_i 数量是 t 的成倍指数级的, 有两个主要利用方法, 首先公式 (15.15) 的最大值能被更有效地演算: 近似地通过替换最大值 [Emery-Montemerlo et al, 2004] 或者确切地通过启发式搜索 [Kumar and Zilberstein, 2010b; Oliehoek et al, 2010]。其次, 通过剪枝 [Emery-Montemerlo et al, 2004]、近似聚类 [Emery-Montemerlo et al, 2005] 或无损聚类 [Oliehoek et al, 2009] 减少关注的历史是可能的。

487

15.4.4.3 多学习器 A^*

由于运用 \hat{Q} 的 FSPC 能看成一棵搜索树中的一条路径, 自然的想法就是允许回溯并向多学习器 $A^*(MAA^*)$ (多学习器动作集) 那样进行完全启发式搜索 [Szer et al, 2005], 如图 15.7b 所示。

① 由于 Q^* 是关于 φ_i 和 δ_i 的函数, 公式 (15.13) 和公式 (15.15) 有轻微的不同, 前者在技术上和 CBG (协作贝叶斯博弈) 无关, 而后者却有关。

② $Q^*(\bar{\theta}_i, \mathbf{a})$ 和 $Q^*(I, \varphi_i, \bar{\theta}_i, \delta_i)$ 有一个微小却很重要的不同就是: 后者指定了给定的任何过去联合策略 φ_i 的最优值, 而前者则仅仅指定了假设 π^* 被实际跟随时最优值。

MAA* 表示一个 A* 通过过去联合策略 φ_t 进行搜索, 通过 $V^{0 \dots t-1}(\varphi_t)$ 、开始阶段 t 的实际期望价值, 加上 $h-t$ 阶段的启发值 $\hat{V}^{t \dots h-1}$ 来计算启发值 $\hat{V}(\varphi_t)$, 当启发式方法可接受 (肯定是高估的), 所以就是 $\hat{V}(\varphi_t)$ 。MAA* 演示了标准 A* 搜索 [Russell and Norvig, 2003]: 它维护局部联合策略 φ_t 和启发值 $\hat{V}(\varphi_t)$ 的公开表 P, 在每一次迭代中 MAA* 选择并扩展排名最高的 φ_t , 产生并启发式地评估所有的 $\varphi_{t+1} = \langle \varphi_t \circ \delta_t \rangle$ 并把它们放入 P, 当使用一个可接受的启发式方法时, 新扩展策略的启发式值 $\hat{V}(\varphi_{t+1})$ 是真实价值以及目前已发现的、能被用于修剪 P 的下界值 \underline{v}^* 的上界, 当列表 P 为空时, 搜索结束, 此时被完全指明的联合策略的最优解就找到了。

MAA* 和之前小节中描述的最优价值函数有直接的联系: 公式 (15.14) 中给出的 V^* 就是最优启发解 $\hat{V}^{t \dots h-1}$ (注意 V^* 值只指明阶段 t 前面的奖励)。

通过 CBG 模型得到的 MAA* 和 FSPC 有相同难题: δ_t 的数量以 t 的成倍指数级增长, 这意味着一个节点的子树以成倍指数级增长。为了缓解这个问题, 利用无损聚类 [Oliehoek et al, 2009] 或尝试让所有子节点只在需要时增长节点 [Spaan et al, 2011] 都是可能的。

15.4.4.4 广义的 MAA*

尽管图 15.7 展示了 FSPC 和 MAA* 之间清晰的关系, 它们到底怎样联系的还不明确: 前者解决了协作贝叶斯博弈问题, 后者则演示了启发式搜索, 广义的 MAA* (GMAA*) [Oliehoek et al, 2008b] 通过明确 “扩展” (Expand) 器将两种方法结合起来。

算法 26 演示了 GMAA*, 当选择 (Select) 器选择排名最高的 φ_t 且扩展器像 MAA* 描述的那样时, GMAA* 单单只是 MAA*, 或者, 扩展器可以为所有被评估的联合 CBG 策略 δ_t 构建 CBG $B(I, \varphi_t)$ 。这些可以用于构建一组新的部分策略集 $\Phi_{\text{Expand}} = \{ \langle \varphi_t \circ \delta_t \rangle \}$ 及其启发式值。这对 MAA* 相当于改编为在 CBG 中工作。它可以显示当使用 \hat{Q} 的特定形式时 (包括上述启发式 Q_{MDP} 、 Q_{POMDP} 和 Q_{BG}), 方法是相同的 [Oliehoek et al, 2008b]。GMAA* 也可以使用一个不构造新的部分策略的扩展器, 但是只用最好的一个, $\Phi_{\text{Expand}} = \{ \langle \varphi_t \circ \delta_t^* \rangle \}$, 结果开放列表 P 将不包含多个部分策略, 并且动作减少到 FSPC。另外, 称为 k -GMAA* 的分组构造 k 个最佳排名的部分策略, 允许折衷计算时间和解决方案质量。历史的聚类也可以应用于 GMAA*, 但只有无损聚类将保持最优性。

```

Initialize:  $\underline{v}^* \leftarrow -\infty$ ,  $P \leftarrow \{\varphi_0 = ()\}$ 
repeat
   $\varphi_t \leftarrow \text{Select}(P)$ 
   $\Phi_{\text{Expand}} \leftarrow \text{Expand}(I, \varphi_t)$ 
  if  $\Phi_{\text{Expand}}$  contains full policies  $\Pi_{\text{Expand}} \subseteq \Phi_{\text{Expand}}$  then
     $\pi' \leftarrow \arg \max_{\pi \in \Pi_{\text{Expand}}} V(\pi)$ 
    if  $V(\pi') > \underline{v}^*$  then
       $\underline{v}^* \leftarrow V(\pi')$  {found new lower bound}
       $\pi^* \leftarrow \pi'$ 
       $P \leftarrow \{\varphi \in P \mid \hat{V}(\varphi) > \underline{v}^*\}$  {prune P}
       $\Phi_{\text{Expand}} \leftarrow \Phi_{\text{Expand}} \setminus \Pi_{\text{Expand}}$  {remove full policies}
     $P \leftarrow (P \setminus \varphi_t) \cup \{\varphi \in \Phi_{\text{Expand}} \mid \hat{V}(\varphi) > \underline{v}^*\}$  {remove processed/add new  $\varphi$ }
until P is empty

```

算法 26 (一般化) MAA*

15.4.5 后推法: 动态规划

Dec-POMDP 的前推法为初始阶段 $t=0$ 和最后阶段 $t=h-1$ 逐步建立策略, 在这之前, 需

要计算一个 Q 值函数 (最优值 Q^* 或近似值 \hat{Q})，计算过程本身 (即动态规划) 由定理 15.2 给出，从最后阶段开始然后反向执行。最终的最优值和联合决策规则和后向持续优化的期望值相关。即，从公式 (15.10) 看出，这能解释为像计算最优联合子树策略的子集的价值那样。

这一节将用动态规划来求解 Dec-PODMP [Hansen, Bernstein, and Zilberstein, 2004]，这种方法同样是有反向作用，但不是计算 Q 值函数，而是直接计算有用子树策略集。

489

15.4.5.1 Dec-POMDP 的动态规划

DP 的核心思想是为学习器逐步建立更长子树策略集合：从上一个阶段执行的 1 步到达 ($\tau=1$) 的子树策略集 (动作) 开始，建立一个在阶段为 $h-2$ 时执行的 2 步到达的策略，即，DP 为所有的学习器 i 构造 $Q_{\tau=1}^i, Q_{\tau=2}^i, \dots, Q_{\tau=h}^i$ ，当上一个备份步骤完成，由于 15.3.4 节中描述的初始信任值 I ，最优策略能通过评估所有的诱发的联合策略 $\pi \in Q_{\tau=h}^1 \times \dots \times Q_{\tau=h}^n$ 找到。

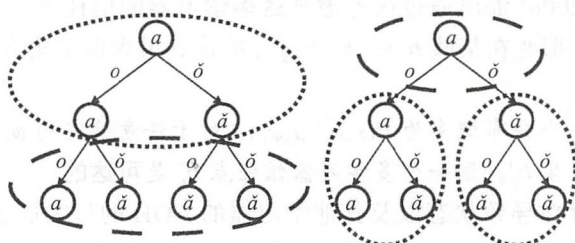


图 15.8 一个动作值为 a, \tilde{a} 以及观测值为 o, \tilde{o} 的学习器的 MAA* (左边) 和动态规划 (右边) 创建策略的不同点。虚线部分是新生成的，带点的部分为前一个迭代的结果

DP 使用建立 $Q_{\tau=k+1}^i$ 到 $Q_{\tau=k}^i$ 的备份操作的想法形式化，例如，图 15.8 的右侧展示了 $q_{\tau=3}^i$ (一个 3 步到达子树策略)，从 $q_{\tau=2}^i \in Q_{\tau=2}^i$ 创建，通俗的说，一个 1 步扩展策略 $q_{\tau=k+1}^i$ 通过为根的每一个观测值和动作值选择子树策略来创建，一个完全的备份产生所有可能的、像它们的子树之前产生集中的策略 $q_{\tau=k+1}^i \in Q_{\tau=k}^i$ ，我们将用 $Q_{\tau=k+1}^{e,i}$ 表示所有学习器 i 由彻底备份产生的子树策略集。

不幸的是，子树策略集以 k 的成倍指数级持续增长[⊖]，为了应付这个难解决的问题，从 $Q_{\tau=k}^{e,i}$ 修剪支配子树策略是可能的，产生更小的持续集 $Q_{\tau=k}^{m,i}$ [Hansen et al, 2004]。 $q_{\tau=k}^i$ 的值依赖于开始各状态分布的概率以及其他学习器选择它们子树策略的概率。因此，如果在多学习器信任空间任一点 $q_{\tau=k}^i$ 都不是最大值，那它就是可支配项：单纯形 $S \times Q_{\tau=k}^{m,-i}$ 。通过线性规划测试可支配项是可能的，删除一个学习器 i 的支配子树策略 $q_{\tau=k}^i$ 可能导致另一个学习器 j 的 $q_{\tau=k}^j$ 成为控制项，因此 DP 对学习器进行迭代直到不再修剪是可能的，这就是人们知道的迭代消除劣势策略 [Osborne and Rubinstein, 1994]。

490

在实践中，DP 的修剪步骤通常不能通过减少持续集来使更大的问题变得易处理。有限 DP 能用于计算一个边界近似值 [Amato et al, 2007]。它执行更具侵略性的 ϵ 修剪：在一些多学习器信任空间中是最大的 $q_{\tau=k}^i$ 也被修剪，其至多通过 ϵ 提高这个区域中的值。因为使用 ϵ 修剪消除迭代仍然会导致价值的无限减少，BDP 执行一次 ϵ 修剪迭代，紧跟的就是通过使用正常修剪消除迭代。

很多子树策略都能被修剪，DP 可能在完全备份时遇到问题，渐增策略代数 (IPG) 是一个通过执行 1 步状态可达性分析来解决问题的技术 [Amato et al, 2009]。在为学习器 i 备份子

⊖ 由于 $q_{\tau=k}^i$ 是 k 维问题重要的全策略，它们的数量以 k 的成倍指数级增长。

树时, IPG 为每一个 $\langle a^i, o^i \rangle$ 对分析一组有非零概率的状态 $S_{\langle a^i, o^i \rangle}$ (一个特定观测值可能不包括许多状态)。接着, 在构造集合 $Q_{\tau=k+1}^{e,i}$ 时, 只有没被控制的子树策略 $S_{\langle a^i, o^i \rangle}$ 才被选为动作 a^i 和观察 o^i , 这能产生更小的子树策略集。

DP 的另一个难题就是为了执行修剪, 所有的 $V(s, q_{\tau=k})$ 值需要被计算和储存, 花费 $|Q_{\tau=k}^{e,i}|^n \times |S|$ 。像这样, DP 在缺乏时间之前就缺少内存了, 为了解决这个问题, [Boularias and Chaib-draa, 2008] 提出通过利用顺序形式 [Koller et al, 1994] 表示法更细密地表示这些值, 然而, 其中一个缺点就是这个方法会导致可支配策略的保持。像这样, 就要求有存储所有子树策略的值和数量的空间。

15.4.5.2 基于点的 DP

DP 仅移除在多学习器信任空间中的任何点处都不是最大的 $q_{\tau=k}^i$ 。基于点的 DP (PBDP) [Szer and Charpillet, 2006] 提出通过仅考虑可达多学习器的信任点子集 $\mathcal{B}^i \subset \mathcal{P}(S \times Q_{\tau=k}^i)$ 改进修剪集合 $Q_{\tau=k}^{e,i}$, 只有那些在某些 $b^i \in \mathcal{B}^i$ 下 $q_{\tau=k}^i$ 能达到最大值的集合才被保留。下面简单提及可达的定义。

定义 15.9 如果存在一个分布概率为 $P(s, \bar{\theta}_t^{-i} | I, \varphi_t)$ (对于任意确定的 φ_t), 且用 $\Gamma_t^i: \bar{\theta}_t^i \rightarrow Q_{\tau=k}^i$ 映射至 $\Gamma_t^{-i} = \langle \Gamma_t^j \rangle_{j \neq i}$ 产生 b_t^i , 则一个多学习器信任点 b_t^i 是可达的。

也就是说, 如果存在导致状态以及其他学习器的 AOH 的过去联合策略近似分布的点, 则该信任点 b^i 是可达的, 当和一个映射到子树策略的 AOH 相结合时, b^i 是状态和子树策略的最终分布。

根据公式 (15.10) 来理解 PBDP, 假设 Γ_t^i 的范围被限制为由完全备份产生的集合: $\Gamma_t^i: \bar{\theta}_t^i \rightarrow Q_{\tau=k}^{e,i}$, 为了一个过去联合策略求解公式 (15.10), 将导致给定 φ_t 时, Γ_t^* 为所有学习器指定所有有用的子树策略 $q_{\tau=k}^i \in Q_{\tau=k}^{e,i}$, 为所有 φ_t 求解公式 (15.10) 将产生所有潜在有用的集合 $q_{\tau=k}^i \in Q_{\tau=k}^{e,i}$ 。

给定一个 φ_t 和 Γ_t^{-i} , 公式 (15.10) 从学习器 i 的角度说明最优返回值能被重写为最大值^①:

$$BR^i(\bar{\theta}_t^i, \varphi_t, \Gamma_t^{-i}) = \arg \max_{q_{\tau=k}^i} \sum_{\bar{\theta}_t^i} \sum_{s_t} P(s_t, \bar{\theta}_t^{-i} | \bar{\theta}_t^i, I, \varphi_t) V(s_t, \langle \Gamma_t^{-i}(\bar{\theta}_t^{-i}), q_{\tau=k}^i \rangle) \quad (15.16)$$

也就是说, 给定 φ_t 和 Γ_t^{-i} , 每一个 $\bar{\theta}_t^i$ 产生一个多学习器信任点, 公式 (15.16) 表示其最大值。所有 φ_t 、 Γ_t^{-i} 和 $\bar{\theta}_t^i$ 的最优返回值集合 $Q_{\tau=k}^{m,i} := \{BR^i(\bar{\theta}_t^i, \varphi_t, \Gamma_t^{-i})\}$ 包含了所有非支配子树策略, 因此产生一个确切的修剪步骤。

PBDP 用初始信任来克服对支配整个多学习器信任空间的测试问题, 这也有可能导致更多的剪枝操作, 因为它避免保持部分不可达空间最大化的子树策略。上述操作还是难解的, 因为 $(\bar{\theta}_t^i, \varphi_t, \Gamma_t^{-i})$ 的数量是 t 的成倍指数级, 且持续集 $Q_{\tau=k}^{m,i}$ 仍能呈成倍指数级增长。

15.4.5.3 内存有界 DP

内存有界 DP (MBDP) [Seuken and Zilberstein, 2007b] 是一种通过产生两个近似值解决这些复杂问题的近似方法, 第一个近似值是假设一个使联合信任值达到最大的联合子树策略可能指明一个好的候选个别子树策略, 而不是像公式 (15.16) 那样来计算候选子树策略:

$$\forall_{\bar{\theta}_t^i} \quad q_{\tau=k}^{\bar{\theta}_t^i} = \arg \max_{q_{\tau=k}^e} V(I, \bar{\theta}_t^i, q_{\tau=k}) \quad (15.17)$$

其中 $Q_{\tau=k}^e \triangleq Q_{\tau=k}^{e,1} \times \cdots \times Q_{\tau=k}^{e,n}$ 是由彻底备份子树 $Q_{\tau=k}^{e,i}$ 产生的集合 $q_{\tau=k}$, 如果 $q_{\tau=k}^i$ 不是任意 $q_{\tau=k}^{\bar{\theta}_t^i}$

① 状态的合计值通过替换掉公式 (15.7) 中的 $V(I, \bar{\theta}_t, q_{\tau=k})$ 得出。

的部分, 则假设它是可支配的。注意公式 (15.17) 中定义的 $V(I, \bar{\theta}_t, q_{\tau=k})$, 通过它产生的联合信任值 b 仅依赖于 $\bar{\theta}_t$, 所以公式 (15.17) 仅被不同的 b 评估, 而且注意这个最大值是意料之外的: 这在 15.4.3.1 节中说明了演示一个 AOH 树的特定节点的最大化过程是不可能的。这里的区别是 MBDP 将不会使用已知的 $q_{\tau=k}^{\bar{\theta}_t}$ 作为 $\bar{\theta}_t$ 的联合子树策略 (这可能产生一个集中式的联合策略), 而是利用它构建个别候选集。 $q_{\tau=k}^i$

492

第二个近似值是 MBDP 维持一个固定大小为 M 的集合 $Q_{\tau=k}^{m,i}$, 有两个主要的结果。第一, 完全备份形成的候选集 $Q_{\tau=k}^{e,i}$ 的大小是 $O(|A^\dagger|M^{|\mathcal{O}^\dagger|})$, 它不依赖于时间, 第二, 公式 (15.17) 不需要评估所有不同的 b , 不像公式 (15.17) 评估 MBDP 样本 M 的联合信任点 b 那样[⊖]。为了演示这个观测过程, MBDP 使用启发式策略。

为了在公式 (15.17) 中算出最大值, MBDP 为每一个样本信任点循环执行 $|Q_{\tau=k}^e|=O(|A^\dagger|M^{|\mathcal{O}^\dagger|})$ 求联合子树策略。为了降低复杂性, 一些论文已经提出新的执行基于点备份操作的方法 [Seuken and Zilberstein, 2007a; Carlin and Zilberstein, 2008; Dibangoye et al, 2009; Amato et al, 2009; Wu et al, 2010a], 这个备份与解决每一个联合动作的 CBG 问题相关 [Kumar and Zilberstein, 2010b; Oliehoek et al, 2010]。

最后, 基于样本的扩展已经被提出 [Wu et al, 2010c,b], 这些都利用样本来评估数量 $V(s, q_{\tau=k})$, 并用部分代表联合样本信任值。

15.4.6 其他有限域的方法

还有一些其他方法用于求解有限区间的 Dec-POMDP 问题, 我们将只在这里简要描述。[Aras et al, 2007] 提出了一种混合整数线性规划方法, 基于用序列形式 [Koller 和 Pfeffer, 1997] 表示每个学习器的可能策略集求解有限 Dec-POMDP 的最优解。在该表示中, 学习器 i 的策略表示为序列集的子集 (大致对应于动作观测历史)。这样的问题可以解释为组合优化问题并用混合整数线性规划求解。

求解 Dec-POMDP 可以作为组合优化问题看待, 同时也可以通过基于交叉熵的优化方法 [Oliehoek et al, 2008a] 和遗传算法 [Eker and Akın, 2008] 来解决。

15.5 延伸阅读

本节给一些深层次的 Dec-POMDP 话题提了一些建议。

15.5.1 一般化和特殊问题

Dec-POMDP 的一般化就是部分可观察随机博弈模型 (POSG), 它与 Dec-POMDP 有相同的成分, 不包括它指定了一个奖励函数的集合: 每个学习器都有一个奖励函数。POSG 假定使个人预期累计奖励达到最大的利己主义的学习器, 不再是仅有一个最佳联合政策的简单概念, 而是联合策略应为纳什均衡 (NE)、并且最好是最优的纳什均衡 (NE)。然而, 却没有一个明确的方式来定义什么是最优的。此外, 这样的 NE 仅能保证存在于随机策略中 (对于有限 POSG), 这意味着它不再执行穷举策略评估。而且, 基于交替最大化的搜索方法也不能保证 POSG 收敛。在 15.4.5.1 节中讨论的 (不是基于点的) 动态规划方法, 适用于 POSG, 因为它为每个学习器找到一组非支配策略。

493

⊖ 如果公式 (15.17) 的评估导致 $q_{\tau=k}^i$ 的重复, 则需要更多的样例。

由于 Dec-POMDP 的负复杂性结果, 大量研究已经着重于下面给出线索的特殊情况。关于更多综合概述, 请参考文献 [Pynadath and Tambe, 2002; Goldman and Zilberstein, 2004; Seuken and Zilberstein, 2008]。

一些特殊情况是由不同程度的可观测性形成的, 这些范围从完全或单独可观测 (如多学习器 MDP [Boutilier, 1996]) 到不可观测。在不可观测的情况下, 学习器使用开环策略并且从复杂性的角度来看更容易解决 (NP 完全问题 [Pynadath and Tambe, 2002])。在这两个极端之间有部分可观察的问题。还发现了一个特例, 即联合观测的情况, 其不是单独可观测的, 而是通过联合观测识别真实状态。一个联合可观测 Dec-POMDP 称为 Dec-MDP, 它是拥有 NEXP 完全性结果的 Dec-MDP 的复杂子类 [Bernstein et al, 2002]。

其他研究试图利用状态、转换和奖励的结构。例如, 许多方法都是基于分解 Dec-POMDP 的特殊情况。分解的 Dec-POMDP [Oliehoek et al, 2008c] 是状态空间被分解的 Dec-POMDP, 即状态 $s = \langle x_1, \dots, x_k \rangle$ 被指定为一些状态变量或要素的赋值。对于分解的 Dec-POMDP, 转换和奖励模型通常可以通过更紧凑的贝叶斯网络和附加的奖励来分解 (总奖励是在学习器子集上指定的多个“较小”奖励函数的总和)。许多特殊情况试图通过将状态要素集分割为每个学习器的单独状态 s_i 来利用学习器之间的独立性。

一个与转换和观测无关 (TOI) Dec-MDP [Becker et al, 2004b; Wu and Durfee, 2006] 的例子假定每个学习器 i 具有它自己的局部状态 s_i 和转换的 MDP, 但是这些 MDP 通过某些事件中的奖励函数后是耦合的: 联合动作和联合状态的一些组合将导致额外的奖励 (或惩罚)。这个工作说明了为计算策略 π^j 的最佳返回值, 学习器 i 可能不需要对所有 π^j 的细节进行推理, 但可以使用更抽象的 π^j 表示其对自身的影响。这个核心思想也用于事件驱动 (ED) 的 Dec-MDP [Becker et al, 2004a], 建立的模型中奖励是独立的, 但有某些事件导致转换依赖。[Mostafa and Lesser, 2009] 介绍了 EDI-CR, 一种使 TOI 和 ED-Dec-MDP 一般化的类型。最近抽象的想法已经由 [Witwicki and Durfee, 2010] 进一步探索, 产生了更一般的基于影响的分解 Dec-POMDP 的更一般子类的策略抽象, 称为时间解耦 Dec-POMDP (TD-POMDP), 也使 TOI 和 ED-Dec-MDP [Witwicki, 2011] 一般化。虽然比 TOI Dec-MDP (例如, 学习器的本地状态可以重叠) 更一般, 但 TD-POMDP 仍然是限制性的, 因为它不允许多个学习器对相同的状态因子具有直接影响。

最后, 已经有关于网络分布式 POMDP (ND-POMDP) 的一系列研究 [Nair et al, 2005; Kim et al, 2006; Varakantham et al, 2007; Marecki et al, 2008; Kumar and Zilberstein, 2009; Varakantham et al, 2009]。ND-POMDP 可以通过 TOI 和其他的分解奖励函数理解为分解的 Dec-POMDP。该模型显示出了价值函数 $V(\pi)$ 可以被叠加分解。因此, 有可能应用许多分布式约束优化以便更有效地优化价值。像这样的 ND-POMDP 已经成为数量合适 (高达 20) 的学习器。这些结果由 [Oliehoek, 2008c] 扩展到一般的分解 Dec-POMDP。在这种情况下, 独立的程度取决于过程所处阶段; 早期阶段通常是完全耦合的, 限制为小范围和少数 (3 个) 学习器的精确解决方案。然而, 近似解决方案已证明可提供给数以百计规模的学习器。

15.5.2 有限 Dec-POMDP

本章的主要重点是找到有限 Dec-POMDP 的解决方案。在有限 Dec-POMDP 问题上也有相当多的研究, 其中一些在这里总结。

无限域的情况与有限时间域情况有很大不同。例如, 无限 Dec-POMDP 问题是不可判定

的 [Bernstein et al, 2002], 这是 (单个学习器) 无限 POMDP 不可判定性的直接结果 [Madani et al, 1999]。可以通过思考来理解策略表示; 在无限区间情况下, 策略树本身应该是范围无限的, 并且无法在有限的内存内清楚地表示。

因此, 对无限 Dec-POMDP 的研究集中在用限定性策略表示的近似算法中, 常见的选择就是状态控制器 (FSC)。限制策略存储器数量的一个副作用就是在许多情况下允许随机策略可能是有益的 [Singh et al, 1994]。这一工作领域的大多数研究都提出了逐步提高控制器的质量的方法。例如, [Bernstein et al, 2009] 提出策略迭代算法, 通过迭代计算 ε 最优解, 在 FSC 上形成备份操作。但是, 这些备份会以指数级增加控制器的大小, 尽管保留值的变换可以缩小控制器的大小, 但控制器仍然可以无限增长。

495

克服这个问题的一个思路是 POMDP 的有限策略迭代器 (BPI) [Bernstein et al, 2005]。BPI 通过应用有限备份保持 FSC 的节点数固定。给定特定的控制器大小使 BPI 收敛到局部最优。[Amato et al, 2010] 也考虑在给定特定的控制器大小时寻找最佳的联合策略, 而提出非线性规划 (NLP) 公式。虽然这个公式表征真正的最优, 但是, 求解 NLP 确实是困难的, 然而, 近似 NLP 求解器已经在实践中展示了好的结果。

最后, 最近的一个发展解决非限定性 Dec-POMDP 问题是通过规划推理范式 [Kumar and Zilberstein, 2010a]。[Pajarinen and Peltonen, 2011] 将这种方法扩展到了分解 Dec-POMDP 问题。

15.5.3 强化学习

下一个相关的问题是多学习器强化学习 (MARL) 的更一般建模, 也就是说, 本章主要介绍给定模型的规划任务。然而, 在 MARL 建模中, 学习器不能达成这样的模型。相反, 模型必通过在线学习获得 (基于模型的 MARL) 或学习器将必须使用无模型的方法。虽然有大量关于 MARL 的一般工作 [Buşoniu et al, 2008], 但 MARL 在像 Dec-POMDP 这样的建模问题中却很受关注。

很难在这些部分可观察的环境中适当地定义多学习器的 RL 问题的设置, 这可能是文学上产生这种差距的主要原因之一。例如, 不清楚学习器何时或如何观测奖励值^①。此外, 即使当学习器可以观测状态, MARL 的收敛还不是很清楚: 从一个学习器的角度来看, 环境已经变得不稳定, 因为其他学习器也在学习, 这意味着无法保证单一学习器 RL 的收敛。[Claus and Boutilier, 1998] 认为, 在协作环境中, 独立的 Q 学习器保证收敛到局部最优, 但不是最优解。但是, 这种方法在实践中有时被报道是成功的 (例如, [Crites and Barto, 1998]), 而且单一学习器收敛的理论学习正在进步 (例如, [Tuyls et al, 2006; Kaisers and Tuyls, 2010; Wunderet al, 2010])。存在收敛到最优解 [Vlassis, 2007] 的耦合学习方法 (例如, 使用联合 Q 学习动作空间)。然而, 所有形式的耦合学习在真正的 Dec-POMDP 建模中被排除: 例如算法需要完全观测其他学习器的状态和动作, 或所有状态信息的通信。

总结本节, 我们将给像 Dec-POMDP 建模中的 RL 提供几个值得注意的建议。[Peshkin et al, 2000] 引入了分散梯度上升策略搜索 (DGAPS), 一种 MARL 的基于梯度下降的部分可观察建模方法。DGAPS 代表单个策略, 并假设学习器观测全局奖励。基于此, 相对于返回值, 每个学习器也可能独立地在梯度方向上更新其策略, 以产生局部最优联合策略。这

496

① 甚至在部分可观察马尔可夫决策过程中, 不假设学习器获得了立即奖励, 因为它们能传达状态的隐藏信息。

个方法已扩展到学习自动配置模块化机器人的策略 [Varshavskaya et al, 2008]。[Chang et al, 2004] 也考虑了分散式 RL, 假设全局奖励可用于学习器。在他们的方法中, 全局奖励被解释为由于其他学习器影响而被破坏的单个奖励。每个学习器明确尝试通过卡尔曼滤波估计单个奖励并使用已过滤的单个奖励表示独立的 Q 学习。[Wu et al, 2010b] 的方法与 RL 密切相关, 因为它不需要一个模型作为输入。然而, 它需要一个可以初始化为特定状态的模拟器。此外, 算法本身是集中化的, 因此不直接适用于在线 RL。

最后, 有的部分可观察分散建模的 MARL 方法只需要有限的通信量。例如, [Boyan and Littman, 1993] 考虑了用于分组路由问题的分散式 RL。其方法 (Q 路由) 执行一种 Q 学习, 其中只有有限的本地通信: 邻居节点传递某组的未来期望等待时间。[Chang and Ho, 2004] 将 Q 路由扩展到移动无线网络领域。类似的问题 (分布式任务分配) 由 [Abdallah and Lesser, 2007] 提出。在这个问题上还有一个网络, 但现在学习器不发送通信分组, 而是向邻居发送任务。同样地, 通信只是本地的。最后, 在多学习器 MDP 的一些 RL 方法中 (即, 耦合方法), 如果学习器有本地通信的能力, 学习器就可能观测一个子集的状态因子 [Guestrin et al, 2002; Kok and Vlassis, 2006]。

15.5.4 通信

Dec-POMDP 已经扩展到明确地将通信动作和观测组合起来。最终的模型 Dec-POMDP-Com [Goldman and Zilberstein, 2003, 2004] 包括可以由每个学习器发送的一组消息以及指明发送每条消息花费成本的成本函数。Dec-POMDP-Com 的一个目标是:

“找到一个联合策略, 在有限区间内使预期总奖励最大化。该策略的求解嵌入了通信消息的最佳定义” [Goldman and Zilberstein, 2003]。

也就是说, 在这个透视图中, 通信动作的语义成为优化问题的一部分 [Xuan et al, 2001; 497 Goldman and Zilberstein, 2003; Spaan et al, 2006; Goldman et al, 2007]。

还可以考虑消息具有固定语义的情况。在这种情况下, 学习器需要一种机制来处理这些语义。例如, 当学习器共享它们的局部观测值时, 每个学习器维持联合信任值, 并形成这种联合信任的更新, 而不是维持观测值列表。[Pynadath and Tambe, 2002] 表明, 在免费通信下, 每个阶段共享局部观测值的联合通信策略是最优的。许多研究已经证明了在和 Dec-POMDP-Com 相似的模型中共享局部观测值 [Ooi and Wornell, 1996; Pynadath and Tambe, 2002; Nair et al, 2004; Becker et al, 2005; Roth et al, 2005b, a; Spaan et al, 2006; Oliehoek et al, 2007; Roth et al, 2007; Goldman and Zilberstein, 2008; Wu et al, 2011]。

最后一点是, 尽管显式通信模型似乎比隐式通信模型更泛化, 但可以将前者转换为后者。也就是说, Dec-POMDP-Com 可以转换为 Dec-POMDP [Goldman and Zilberstein, 2004; Seuken and Zilberstein, 2008]。

致谢。感谢 Leslie Kaelbling 和 Shimon Whiteson 提供的有价值的反馈、评论家们有远见的评论以及 AFOSR MURI 编号为 #FA9550-09-1-0538 项目的支持。

参考文献

- Abdallah, S., Lesser, V.: Multiagent reinforcement learning and self-organization in a network of agents. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 172–179 (2007)

- Amato, C., Carlin, A., Zilberstein, S.: Bounded dynamic programming for decentralized POMDPs. In: Proc. of the AAMAS Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains, MSDM (2007)
- Amato, C., Dibangoye, J.S., Zilberstein, S.: Incremental policy generation for finite-horizon DEC-POMDPs. In: Proc. of the International Conference on Automated Planning and Scheduling, pp. 2–9 (2009)
- Amato, C., Bernstein, D.S., Zilberstein, S.: Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs. *Autonomous Agents and Multi-Agent Systems* 21(3), 293–320 (2010)
- Aras, R., Dutech, A., Charpillet, F.: Mixed integer linear programming for exact finite-horizon planning in decentralized POMDPs. In: Proc. of the International Conference on Automated Planning and Scheduling (2007)
- Becker, R., Zilberstein, S., Lesser, V.: Decentralized Markov decision processes with event-driven interactions. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 302–309 (2004a)
- Becker, R., Zilberstein, S., Lesser, V., Goldman, C.V.: Solving transition independent decentralized Markov decision processes. *Journal of Artificial Intelligence Research* 22, 423–455 (2004b)
- Becker, R., Lesser, V., Zilberstein, S.: Analyzing myopic approaches for multi-agent communication. In: Proc. of the International Conference on Intelligent Agent Technology, pp. 550–557 (2005)
- Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research* 27(4), 819–840 (2002)
- Bernstein, D.S., Hansen, E.A., Zilberstein, S.: Bounded policy iteration for decentralized POMDPs. In: Proc. of the International Joint Conference on Artificial Intelligence, pp. 1287–1292 (2005)
- Bernstein, D.S., Amato, C., Hansen, E.A., Zilberstein, S.: Policy iteration for decentralized control of Markov decision processes. *Journal of Artificial Intelligence Research* 34, 89–132 (2009)
- Boularias, A., Chaib-draa, B.: Exact dynamic programming for decentralized POMDPs with lossless policy compression. In: Proc. of the International Conference on Automated Planning and Scheduling (2008)
- Boutilier, C.: Planning, learning and coordination in multiagent decision processes. In: Proc. of the 6th Conference on Theoretical Aspects of Rationality and Knowledge, pp. 195–210 (1996)
- Boyan, J.A., Littman, M.L.: Packet routing in dynamically changing networks: A reinforcement learning approach. In: *Advances in Neural Information Processing Systems*, vol. 6, pp. 671–678 (1993)
- Buşoniu, L., Babuška, R., De Schutter, B.: A comprehensive survey of multi-agent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 38(2), 156–172 (2008)
- Carlin, A., Zilberstein, S.: Value-based observation compression for DEC-POMDPs. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 501–508 (2008)
- Chang, Y.H., Ho, T.: Mobilized ad-hoc networks: A reinforcement learning approach. In: *Proceedings of the First International Conference on Autonomic Computing*, pp. 240–247 (2004)
- Chang, Y.H., Ho, T., Kaelbling, L.P.: All learning is local: Multi-agent learning in global reward games. In: *Advances in Neural Information Processing Systems*, vol. 16 (2004)
- Claus, C., Boutilier, C.: The dynamics of reinforcement learning in cooperative multiagent systems. In: Proc. of the National Conference on Artificial Intelligence, pp. 746–752 (1998)
- Cogill, R., Rotkowitz, M., Roy, B.V., Lall, S.: An approximate dynamic programming approach to decentralized control of stochastic systems. In: Proc. of the 2004 Allerton Conference on Communication, Control, and Computing (2004)

- Crites, R.H., Barto, A.G.: Elevator group control using multiple reinforcement learning agents. *Machine Learning* 33(2-3), 235–262 (1998)
- Dibangoye, J.S., Mouaddib, A.I., Chai-draa, B.: Point-based incremental pruning heuristic for solving finite-horizon DEC-POMDPs. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 569–576 (2009)
- Eker, B., Akin, H.L.: Using evolution strategies to solve DEC-POMDP problems. *Soft Computing - A Fusion of Foundations, Methodologies and Applications* (2008)
- Emery-Montemerlo, R., Gordon, G., Schneider, J., Thrun, S.: Approximate solutions for partially observable stochastic games with common payoffs. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 136–143 (2004)
- Emery-Montemerlo, R., Gordon, G., Schneider, J., Thrun, S.: Game theoretic control for robot teams. In: *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 1175–1181 (2005)
- Goldman, C.V., Zilberstein, S.: Optimizing information exchange in cooperative multi-agent systems. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 137–144 (2003)
- Goldman, C.V., Zilberstein, S.: Decentralized control of cooperative systems: Categorization and complexity analysis. *Journal of Artificial Intelligence Research* 22, 143–174 (2004)
- Goldman, C.V., Zilberstein, S.: Communication-based decomposition mechanisms for decentralized MDPs. *Journal of Artificial Intelligence Research* 32, 169–202 (2008)
- Goldman, C.V., Allen, M., Zilberstein, S.: Learning to communicate in a decentralized environment. *Autonomous Agents and Multi-Agent Systems* 15(1), 47–90 (2007)
- Guestrin, C., Lagoudakis, M., Parr, R.: Coordinated reinforcement learning. In: *Proc. of the International Conference on Machine Learning*, pp. 227–234 (2002)
- Hansen, E.A., Bernstein, D.S., Zilberstein, S.: Dynamic programming for partially observable stochastic games. In: *Proc. of the National Conference on Artificial Intelligence*, pp. 709–715 (2004)
- Kaisers, M., Tuyls, K.: Frequency adjusted multi-agent Q-learning. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 309–316 (2010)
- Kim, Y., Nair, R., Varakantham, P., Tambe, M., Yokoo, M.: Exploiting locality of interaction in networked distributed POMDPs. In: *Proc. of the AAAI Spring Symposium on Distributed Plan and Schedule Management* (2006)
- Kok, J.R., Vlassis, N.: Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research* 7, 1789–1828 (2006)
- Koller, D., Pfeffer, A.: Representations and solutions for game-theoretic problems. *Artificial Intelligence* 94(1-2), 167–215 (1997)
- Koller, D., Megiddo, N., von Stengel, B.: Fast algorithms for finding randomized strategies in game trees. In: *Proc. of the 26th ACM Symposium on Theory of Computing*, pp. 750–759 (1994)
- Kumar, A., Zilberstein, S.: Constraint-based dynamic programming for decentralized POMDPs with structured interactions. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 561–568 (2009)
- Kumar, A., Zilberstein, S.: Anytime planning for decentralized POMDPs using expectation maximization. In: *Proc. of Uncertainty in Artificial Intelligence* (2010a)
- Kumar, A., Zilberstein, S.: Point-based backup for decentralized POMDPs: Complexity and new algorithms. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 1315–1322 (2010b)
- Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In: *Proc. of the National Conference on Artificial Intelligence*, pp. 541–548 (1999)
- Marecki, J., Gupta, T., Varakantham, P., Tambe, M., Yokoo, M.: Not all agents are equal: scaling up distributed POMDPs for agent networks. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 485–492 (2008)
- Mostafa, H., Lesser, V.: Offline planning for communication by exploiting structured interactions in decentralized MDPs. In: *Proc. of 2009 IEEE/WIC/ACM International Conference*

- on Web Intelligence and Intelligent Agent Technology, pp. 193–200 (2009)
- Nair, R., Tambe, M., Marsella, S.: Role allocation and reallocation in multiagent teams: towards a practical analysis. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 552–559 (2003a)
- Nair, R., Tambe, M., Marsella, S.C.: Team Formation for Reformation in Multiagent Domains Like RoboCupRescue. In: Kaminka, G.A., Lima, P.U., Rojas, R. (eds.) RoboCup 2002: Robot Soccer World Cup VI, LNCS (LNAI), vol. 2752, pp. 150–161. Springer, Heidelberg (2003)
- Nair, R., Tambe, M., Yokoo, M., Pynadath, D.V., Marsella, S.: Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In: Proc. of the International Joint Conference on Artificial Intelligence, pp. 705–711 (2003c)
- Nair, R., Roth, M., Yohoo, M.: Communication for improving policy computation in distributed POMDPs. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 1098–1105 (2004)
- Nair, R., Varakantham, P., Tambe, M., Yokoo, M.: Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In: Proc. of the National Conference on Artificial Intelligence, pp. 133–139 (2005)
- Oliehoek, F.A.: Value-based planning for teams of agents in stochastic partially observable environments. PhD thesis, Informatics Institute, University of Amsterdam (2010)
- Oliehoek, F.A., Vlassis, N.: Q-value functions for decentralized POMDPs. In: Proc. of The International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 833–840 (2007)
- Oliehoek, F.A., Spaan, M.T.J., Vlassis, N.: Dec-POMDPs with delayed communication. In: AAMAS Workshop on Multi-agent Sequential Decision Making in Uncertain Domains (2007)
- Oliehoek, F.A., Kooi, J.F., Vlassis, N.: The cross-entropy method for policy search in decentralized POMDPs. *Informatica* 32, 341–357 (2008a)
- Oliehoek, F.A., Spaan, M.T.J., Vlassis, N.: Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research* 32, 289–353 (2008b)
- Oliehoek, F.A., Spaan, M.T.J., Whiteson, S., Vlassis, N.: Exploiting locality of interaction in factored Dec-POMDPs. In: Proc. of The International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 517–524 (2008)
- Oliehoek, F.A., Whiteson, S., Spaan, M.T.J.: Lossless clustering of histories in decentralized POMDPs. In: Proc. of The International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 577–584 (2009)
- Oliehoek, F.A., Spaan, M.T.J., Dibangoye, J., Amato, C.: Heuristic search for identical payoff Bayesian games. In: Proc. of The International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 1115–1122 (2010)
- Ooi, J.M., Wornell, G.W.: Decentralized control of a multiple access broadcast channel: Performance bounds. In: Proc. of the 35th Conference on Decision and Control, pp. 293–298 (1996)
- Osborne, M.J., Rubinstein, A.: *A Course in Game Theory*. The MIT Press (1994)
- Pajarinen, J., Peltonen, J.: Efficient planning for factored infinite-horizon DEC-POMDPs. In: Proc. of the International Joint Conference on Artificial Intelligence (to appear, 2011)
- Paquet, S., Tobin, L., Chaib-draa, B.: An online POMDP algorithm for complex multiagent environments. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems (2005)
- Peshkin, L.: Reinforcement learning by policy search. PhD thesis, Brown University (2001)
- Peshkin, L., Kim, K.E., Meuleau, N., Kaelbling, L.P.: Learning to cooperate via policy search. In: Proc. of Uncertainty in Artificial Intelligence, pp. 307–314 (2000)
- Pynadath, D.V., Tambe, M.: The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research* 16, 389–423 (2002)
- Rabinovich, Z., Goldman, C.V., Rosenschein, J.S.: The complexity of multiagent systems: the price of silence. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 1102–1103 (2003)

- Roth, M., Simmons, R., Veloso, M.: Decentralized communication strategies for coordinated multi-agent policies. In: Parker, L.E., Schneider, F.E., Shultz, A.C. (eds.) *Multi-Robot Systems. From Swarms to Intelligent Automata*, vol. III, pp. 93–106. Springer, Heidelberg (2005a)
- Roth, M., Simmons, R., Veloso, M.: Reasoning about joint beliefs for execution-time communication decisions. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 786–793 (2005b)
- Roth, M., Simmons, R., Veloso, M.: Exploiting factored representations for decentralized execution in multi-agent teams. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 467–463 (2007)
- Russell, S., Norvig, P.: *Artificial Intelligence: A Modern Approach*, 2nd edn. Pearson Education (2003)
- Seuken, S., Zilberstein, S.: Improved memory-bounded dynamic programming for decentralized POMDPs. In: *Proc. of Uncertainty in Artificial Intelligence* (2007a)
- Seuken, S., Zilberstein, S.: Memory-bounded dynamic programming for DEC-POMDPs. In: *Proc. of the International Joint Conference on Artificial Intelligence*, pp. 2009–2015 (2007b)
- Seuken, S., Zilberstein, S.: Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems* 17(2), 190–250 (2008)
- Singh, S.P., Jaakkola, T., Jordan, M.I.: Learning without state-estimation in partially observable Markovian decision processes. In: *Proc. of the International Conference on Machine Learning*, pp. 284–292. Morgan Kaufmann (1994)
- Spaan, M.T.J., Gordon, G.J., Vlassis, N.: Decentralized planning under uncertainty for teams of communicating agents. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 249–256 (2006)
- Spaan, M.T.J., Oliehoek, F.A., Amato, C.: Scaling up optimal heuristic search in Dec-POMDPs via incremental expansion. In: *Proc. of the International Joint Conference on Artificial Intelligence* (to appear, 2011)
- Szer, D., Charpillet, F.: Point-based dynamic programming for DEC-POMDPs. In: *Proc. of the National Conference on Artificial Intelligence* (2006)
- Szer, D., Charpillet, F., Zilberstein, S.: MAA*: A heuristic search algorithm for solving decentralized POMDPs. In: *Proc. of Uncertainty in Artificial Intelligence*, pp. 576–583 (2005)
- Tuyls, K., Hoen, P.J., Vanschoenwinkel, B.: An evolutionary dynamical analysis of multi-agent learning in iterated games. *Autonomous Agents and Multi-Agent Systems* 12(1), 115–153 (2006)
- Varakantham, P., Marecki, J., Yabu, Y., Tambe, M., Yokoo, M.: Letting loose a SPIDER on a network of POMDPs: Generating quality guaranteed policies. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems* (2007)
- Varakantham, P., Young Kwak, J., Taylor, M.E., Marecki, J., Scerri, P., Tambe, M.: Exploiting coordination locales in distributed POMDPs via social model shaping. In: *Proc. of the International Conference on Automated Planning and Scheduling* (2009)
- Varshavskaya, P., Kaelbling, L.P., Rus, D.: Automated design of adaptive controllers for modular robots using reinforcement learning. *International Journal of Robotics Research* 27(3–4), 505–526 (2008)
- Vlassis, N.: *A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence*. In: *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan & Claypool Publishers (2007)
- Witwicki, S.J.: *Abstracting influences for efficient multiagent coordination under uncertainty*. PhD thesis, University of Michigan, Ann Arbor, Michigan, USA (2011)
- Witwicki, S.J., Durfee, E.H.: Influence-based policy abstraction for weakly-coupled Dec-POMDPs. In: *Proc. of the International Conference on Automated Planning and Scheduling*, pp. 185–192 (2010)
- Wu, F., Zilberstein, S., Chen, X.: Point-based policy generation for decentralized POMDPs. In: *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 1307–1314 (2010a)

- Wu, F., Zilberstein, S., Chen, X.: Rollout sampling policy iteration for decentralized POMDPs. In: Proc. of Uncertainty in Artificial Intelligence (2010b)
- Wu, F., Zilberstein, S., Chen, X.: Trial-based dynamic programming for multi-agent planning. In: Proc. of the National Conference on Artificial Intelligence, pp. 908–914 (2010c)
- Wu, F., Zilberstein, S., Chen, X.: Online planning for multi-agent systems with bounded communication. *Artificial Intelligence* 175(2), 487–511 (2011)
- Wu, J., Durfee, E.H.: Mixed-integer linear programming for transition-independent decentralized MDPs. In: Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems, pp. 1058–1060 (2006)
- Wunder, M., Littman, M.L., Babes, M.: Classes of multiagent Q-learning dynamics with epsilon-greedy exploration. In: Proc. of the International Conference on Machine Learning, pp. 1167–1174 (2010)
- Xuan, P., Lesser, V., Zilberstein, S.: Communication decisions in multi-agent cooperation: Model and experiments. In: Proc. of the International Conference on Autonomous Agents (2001)
- Zettlemoyer, L.S., Milch, B., Kaelbling, L.P.: Multi-agent filtering with infinitely nested beliefs. In: *Advances in Neural Information Processing Systems*, vol. 21 (2009)

其他应用领域

强化学习与心理和神经科学之间的关系

Ashvin Shah

摘要

动物行为和心理学的研究对强化学习有很大的启发。早期研究表明，动物可以经过多次训练从而学会做一些（令人满意的）动作，而几十年的后续研究也（并且仍然在继续）旨在发现这种学习的机制。本章首先介绍动物学习中与强化学习（RL）中基本概念直接相关的行为和理论。然后介绍神经科学方面的研究：动物学习所使用的学习机制与许多 RL 算法所使用的机制非常的相似。一直以来，笔者都认为计算机科学方面的研究极大地促进了心理学和神经科学的发展。

16.1 简介

在 19 世纪末期，心理学家爱德华·索恩迪克做了一系列实验，他将一只饥饿的动物（最著名的是一只猫）放在一个“拼图盒”中，门只有在动物执行了一系列特定动作之后才会被打开，如先拉动链子，然后按杠杆。同时让动物发现盒子外边放着一些食物。傻傻的动物一开始会做一些与开门毫不相干的动作，例如撞门、梳理自己。在某些机缘巧合之下，动物可能恰好拉动链子，随后也碰巧按下了杠杆，这时，门就会打开，动物就会从盒子里逃出来，吃掉食物。而当那只动物又被放进盒子里的时候，与开门无关的动作它就做的较少了。

507 经过反复试验，动物最后可以在几秒内从盒子里逃出来。

这个过程是不是很熟悉？没错，它描述的正是强化学习学习器在执行一个简单任务时所采取的动作。动物在拼图盒里所面临的问题也正是强化学习学习器所必须要解决的问题：在没有提示的情况下，学习器应该做什么，什么时候做才能得到比较好的结果？虽然 RL 并不是动物学习的模型，但是动物行为和心理学的研究却促进了 RL 的发展 [Sutton and Barto, 1998]。[Klopf, 1982] 所做的研究也推动了 RL 的发展，他提出享乐动作（“寻求乐趣”）是从享乐学习过程中产生的，包括控制单个神经元的行为过程。

在本章中，首先我会介绍一些早期研究动物行为的实验，沿用至今的一些基本例子和一些心理学理论（用来解释观察到的动作）。然后，我会介绍神经科学的一些研究（与大脑中控制行为的机制有关）。在这一章节中，我们并不是详细介绍动物学习和行为及其背后的神经机制，而是仅仅介绍与 RL 中使用的基本概念直接相关的一些研究和实验方法。我想要大家清楚的一点是：在某些情况下，RL 学习器所使用的机制和动物学习使用的机制是非常相似的。研究动物的行为可以帮助我们解决人工系统的一些问题。这表明，心理学和神经科学的研究可以促进 RL 和机器学习的发展。

16.2 经典 (巴甫洛夫) 条件反射

预测在学习和控制中起着重要的作用。一般的,我们都会用经典条件反射实验来研究预测对学习所起的作用,这个实验由伊万·巴甫洛夫在 20 世纪初期在俄罗斯提出。在研究狗的消化功能的时候,巴甫洛夫注意到参与过实验的狗在食物没有拿出之前就已经开始分泌唾液了。通过尝试解释这一奇怪的现象,巴甫洛夫提出了条件反射理论 [Pavlov, 1927]: 心理过程 (例如,听觉) 可能产生以前被认为仅能由物理过程 (闻到食物或口中存在食物) 引起的生理反应 (唾液分泌)。最出名的是,当饭前总会有铃声时,仅仅铃声就可以让狗分泌唾液。这种行为可以认为是狗已经学会预测铃声之后有食物。

508

16.2.1 行为

虽然巴甫洛夫的流口水实验是条件反射的经典例子,但是深入研究的却是兔子的触诊膜 (NM), 一种闭合后保护眼睛的薄的“第三眼睑”。通常,兔子被控制后,对着眼睛喷气或受到轻微的电击 (无条件的刺激, US) 时会导致 NM 关闭 (无条件反应, UR)。一个中性刺激本身不会使 NM 关闭 (条件刺激, CS), 如音调或光。但是如果总是在 US 之前加一个 CS, 最终 CS 也会使 NM 关闭 (条件反应, CR)。由于 CR 是由 CS 和 US 配对并反复训练获得的 (实验的反应获得阶段), 因此 US 通常称为强化剂。CR 的强度 (通常以 NM 关闭的速度或 US 之前 NM 关闭的概率来衡量) 是用来度量 CS 的预测强度的。在获得 CR 之后, 如果 CS 存在, 但 US 消失了 (消光期), CR 的强度将会逐渐降低。

通过改变实验协议中的一些操作,我们可以更好地了解预测是如何习得的。特别有指导意义的是, CS 相对于 US 出现的时间和使用多个 CS, 都会影响到 CS 预测的质量。[Sutton and Barto, 1987] 对此进行了实验, 提出了经典条件反射的时序差分模型。

如图 16.1a 所示, CS 和 US 之间有两种时间间隔: 1) 刺激间隔 (ISI), 它是 CS 发生和 US 发生之间的时间间隔; 2) 追踪间隔 (TI), 它是 CS 结束和 US 发生之间的时间间隔。比较简单的协议是使用短且恒定的 ISI 和时间间隔为 0 的 TI (延迟性条件反射)。例如, 给兔子听一个短暂的音乐 (500ms) 后, 立即对着兔子眼睛喷气。当 TI 大于零 (追踪性条件反射) 时, CR 的获取和“寿命”会受到一定限制。如果 ISI 为零, 即同时施加 CS 和 US, 则 CS 无法起到预测作用, 并且动物也不会获得 CR。存在一个最优的 ISI (对于 NM 反应大约是 250ms), 超过最优值之后, CR 的强度将会逐渐降低。并且追踪性条件反射比延迟性条件反射下降的速率更快。此外, 随着 ISI 的增加, CR 的获取就会变得困难起来, 这表明预测未来比较遥远的事件是比较困难的。

通过使用多个 CS (复合刺激) 进行实验, 我们发现动物学习也与预测 US 的能力有关。如图 16.1b 所示, 我们把其中一个 CS (例如色调, 称为 CSA) 以浅灰色着色, 而另一个 (例如光, 称为 CSB) 以深灰色着色。在阻滞中, US 单独和 CSA 组合训练并使动物获得 CR。之后, 在 CSA 发生的同时也加入刺激 CSB (使两个刺激同时发生)。随后测试 CSB 是否能使动物获得 CR, 结果发现动物没有获得 CR。在条件抑制中, 使两种机制混合训练: 首先把 CSA 与 US 组合起来训练动物, 随后把这些动物分成两组, 一组会在 CSA 发生时另加入刺激 CSB (使两个刺激同时发生, 复合刺激), 而另一组不加入 CSB, 最后 CSB 单独与 US 配对并对两组动物都进行训练。结果表明经历过复合刺激的动物更容易获得 CR。

在上述两个示例中, 两个 CS 是同时发生的。还有一些实验协议显示了不同时刻发生

的两个 CS 复合起来的结果（连续复合刺激）见图 16.1c）。如前面所述，追踪性条件反射（ $TI>0$ ），会阻碍动物获得 CR。然而如果在 TI 之间加入另一个 CS，则会促进第一个 CS 与 US 形成 CR（通过干预刺激促进）。而在这些实验协议导致高级调节中，则会首先响应 CSB 获得 CR。然后，如果 CSB 之前加入 CSA 的话，则 CSA 也会获得 CR。总而言之，CSB 在这个学习过程中起到强化的作用。

在首因效应中，CSA 和 CSB 在时间上会重合。两个刺激的消失时间是相同的，都发生在 US 发生的前一刻。但是 CSA 开始时间比 CSB 早（见图 16.1c）。因为 CSB 的 ISI 比 CSA 短，所以可以猜出，CSB 与 US 配对比 CSA 与 US 配对更容易使动物获得 CR。然而，在复合刺激中，CSA 的存在却导致了 CR 响应 CSB 强度的降低。更令人惊讶的是 [Sutton and Barto, 1981] 提出的一个预测。他们提出了另一种经典条件反射模型：首先 CSB 与 US 配对进行训练（一个较短的 ISI 延迟性条件反射），然后加入 CSA（ISI 较长）。尽管 CSB 与 CR 关联的强度（代表 CS 的预测能力）已经达到渐近水平。但是加入了 CSA 后，强度依然会下降。这样的结果似乎与 ISI 效应和阻塞现象不一致。[Sutton and Barto, 1987] 重复做了这个实验，但是结果确实如此。[Kehoe et al, 1987] 通过实验也证实了这一预测。

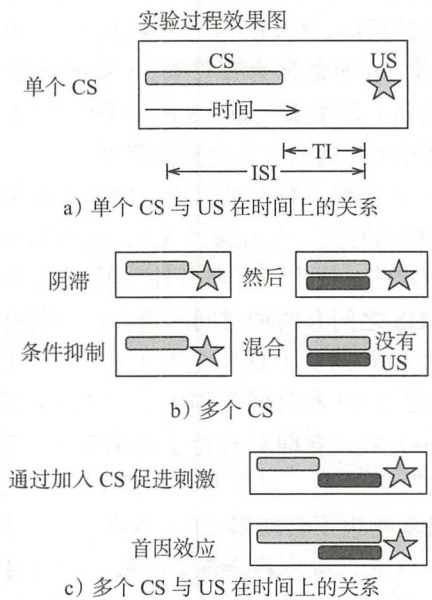


图 16.1 不同类型的经典条件刺激。水平轴线表示时间，星星表示发生 US 的时刻，浅灰色矩形表示 CS 发生和持续时间。深色矩形表示的是另一个 CS

16.2.2 理论

阻滞效应表明，当两件事无关联时，学习才会产生 [Kamin, 1969]。这个效应推动了 Rescorla-Wagner 经典条件反射模型的发展：我们可以使用试验中所有 CS 与 US 联结强度的总和来预测 US 的存在。联结强度随着预测的准确性而变动。对于在实验中出现的每一个 CS i ：

$$\Delta w(i) = \alpha \left(r - \sum_i w(i)x(i) \right)$$

其中 $w(i)$ 是 CS i 与 US 之间的联结强度；如果 CS i 存在，则 $x(i)=1$ ，否则为 $x(i)=0$ ， r 是 US 可以调节的最大量（类似于 US 的“幅度”）， α 是两个 CS 之间的参数（注意，此符号与原始版本不同。）。如果 US 是完全可预测的（即， $r - \sum_i w(i)x(i)=0$ ），则随后加入另一个 CS（初始联结强度为零），总的联结强度将不会增加。这个模型也可以捕获到经典条件反射的几个特征（例如，阻滞）。此外，正如 [Sutton and Barto, 1981] 提到的那样，该模型与 Widrow-Hoff 所使用的学习规则很相似 [Widrow and Hoff, 1960]，它们都表明了预测对学习的重要性。

Rescorla-Wagner 模型是经典条件反射在训练级别的解释：即学习发生在两次训练之间，不会发生在训练中。但是这种模型无法解释刺激在时间上对学习的连续性影响。此外，[Sutton and Barto, 1987] 指出，动物学习过程可能不会用到只存在于试验概念中的一些机制，这些机制本质上有利于隔离事件。

时序差分（TD）模型 [Sutton, 1988 ; Sutton and Barto, 1998] 是在经典条件反射模型 [Sutton and Barto, 1981 ; Barto and Sutton, 1982 ; Sutton and Barto, 1987] 基础上开发出来的，用于实时预测学习结果。在该模型中， r_t 表示在 t 时刻存在 US，在 t 时刻如果 CS i 存在，则 $x_t(i)=1$ ，如果不存在则为 0。 $w_t(i)$ 是在 t 时刻 CS i 和 US 的联结强度。在每个时间点：

$$w_{t+1}(i) = w_t(i) + \alpha \left(r_t + \gamma \left[\sum_i w_t(i)x_t(i) \right]^+ - \left[\sum_i w_t(i)x_{t-1}(i) \right]^+ \right) \bar{x}_t(i)$$

511

其中 γ 是我们熟悉的时间阻尼系数，如果 $\gamma < 0$ ，则 $[\gamma]^+$ 返回零， $\bar{x}_t(i)$ 是资格迹，例如 $\beta \bar{x}_{t-1}(i) + (1-\beta)x_{t-1}(i)$ ，其中 $0 \leq \beta < 1$ 。

在每个时间点，通过调整权重以最小化 $r_t + r[\sum_i w_t(i)x_t(i)]^+$ 和 $[\sum_i w_t(i)x_{t-1}(i)]^+$ 之间的差异（即时序差分误差）。 $r_t + \gamma[\sum_i w_t(i)x_t(i)]^+$ 和 $[\sum_i w_t(i)x_{t-1}(i)]^+$ 是对即将发生的 US 在时间上的预测（准确地讲： $\sum_{k=0}^{\infty} \gamma^k r_{t+k}$ ）。前一个预测包含更多的最新信息（ r_t 和 $x_t(i)$ ），并且可用作后者的目标。资格迹、折扣率和对时间依赖性都需要使模型知道训练过程中刺激之间的相对时间对学习的影响。因为在 t 时刻的预测强化了 $t-1$ 时刻的预测，这样的模型使用到了高级条件反射。

16.2.3 小结和其他注意事项

经典条件反射研究的动作都与动物预测 US 的能力有关。当 CS 和 US 之间的联系强到一定程度时，预测才会产生的。当发生预测误差时（即动物无法预测 CS 后会有 US），预测发生意外时（即大多数时候 CS 后会有 US），有接触时（即少数时候 CS 后没有 US）经过训练，CS 与 US 之间的联系会变强 [Schultz, 2006]。

TD 模型中所使用到的方法大部分是用来捕获经典条件反射实验中观察到的动作的。其他方法的预测仅把实际结果（如，US）当作训练信号的，而 TD 模型中是将实际结果与连续性预测结果的差异（TD 误差）当作训练信号的。这种方法不仅可以解释动物的动作而且还是一种很强大的计算技术 [Sutton, 1988]。

经典条件反射实验还包括：大脑机制如何调节动作，以及当 CS 和 US 之间出现意外时，CR 是如何变化的。为了解释基于神经机制的动作而设计的计算进一步加深了我们对动物如何学习预测的理解 [Gluck, 2008 ; Mirolli et al, 2010 ; Moore and Choi, 1997 ; Brandon et al, 2002]。

最后, UR 和 CR 可以认为是巴普洛夫动作, 因为动物不会自动形成或选择执行它们。但是, US 却可以使动物获得 UR。CR 不是学习得到的, 它是在学习了刺激 (CS) 与最终结果 (US) 之间的联系后产生的。由于动物很少会对环境和动作进行控制, 经典条件反射使我们能够专注于预测。当然, 预测的好处之一是它能更好地利用我们所经历的事情。这是下一节的重点。

16.3 操作性 (工具性) 条件反射

经典条件反射实验表明具有突出刺激 (US) 的动物的动作取决于另一个刺激 (CS), 而与动物执行的动作无关。而操作性条件反射实验则指出突出刺激 (通常是奖励, 如食物) 取决于动物执行的具体动作 [Thorndike, 1991; Skinner, 1938]。(动物被认为是在环境中运作的工具。) 本章开头所描述的基本实验协议构成了本节中描述的大多数实验的基础。

16.3.1 动作

最简单的实验是让动物学习只有一个动作的操作, 如让一只老鼠按一下杠杆或一只鸽子啄一下钥匙。动作通常表示为动作的“强度”, 以其产生的速度、执行的速度或执行的可能性来衡量。操作性条件反射中的基本协议和动作与经典条件反射中的十分相似。在学习期间 (执行某个动作后会有奖励), 动作的强度逐渐增加, 而奖励则作为强化剂作用其中。在学习消退期间 (执行动作之后没有奖励), 动作的强度逐渐降低。

学习还与实验操作者设定的因素有关。在大多数实验中, 动物在进行实验之前处于一个匮乏状态 (例如, 如果奖励是食物, 那么它就处于饥饿状态)。在匮乏状态下, 动作的强度增加得更快, 因此, 匮乏增加了动物学习的“驱动力”或“动机”。其他研究的因素包括奖励的大小 (如, 食物量的增加会加快动物的学习)、动作执行与奖励交付之间的延迟 (如果延迟增加, 动作的强度将会减少)。

影响学习单一动作的因素也会影响到多动作自由选择任务 (如, 有两个杠杆) 中的动作的选择和策略的制定。选择不同的动作会导致不同的结果, 而不同动作的强度通常用相对可能性来衡量 (即选择分布)。而如同大家所想的那样, 动物往往会选择具有较大奖励的和 / 或延迟较少的动作。

我们可以通过测量选择分布来量化一个因素对另一个因素的影响。例如, 假设动作 A 会得到一个恒定大小和延迟的奖励, 动作 B 得到一个即时奖励。我们可以通过不断地减少即时奖励的大小, 使得动物在两个动作之间没有偏好, 这可以用一个时间折扣函数来表示。虽然 RL 中的大多都是使用指数折扣函数, 但是也可以使用双曲线函数研究动作 [Green and Myerson, 2004]。

获得奖励的概率是影响选择分布的另一个因素。在某些任务中, 选择不同的动作, 获得奖励的概率也是不同的。人类和一些动物的动作表明在概率匹配中, 选择分布类似于动作执行的相对概率 [Siegel and Goldstein, 1959; Shanks et al, 2002]。如果总体目标是最大化接收到的总奖励, 这样的策略显然不是最佳的。一些研究表明, 概率匹配可能部分归因于: 任务指令不明确或存在少量与奖励无关的动作 (即操作者可能试图实现除最大化奖励之外的一些目标) [Shanks et al, 2002; Gardner, 1958; Goodnow, 1955]。

笨笨的动物通常不会做实验者想让它们做的一些动作; 必须使用上述的方法和经典条件反射的方法教导动物, 例如, 为了让鸽子可以啄钥匙, 实验者要先让鸽子发现钥匙, 然后给

鸽子食物，鸽子就会啄食物，通过不断训练，当给鸽子钥匙时，鸽子也会啄钥匙了（自我塑造法 [Brown and Jenkins, 1968]）。虽然这种巴普洛夫动作可以指导动物完成某些动作，但是对动物学习其他动作并无帮助 [Dayan et al, 2006]。

如果一个动作可以分步学会，则可以使用塑造法，指导动物通过逐渐改变引起奖励的动作的范围来执行特定的动作 [Eckerman et al, 1980]。例如，为了教鸽子在空中的特定位置啄东西，首先实验者在该位置周围定义了一个大的假想球体。当鸽子碰巧在这个范围内啄东西时，就给鸽子食物。当鸽子一直可以在这个球体内啄东西时，实验者会减小半径，鸽子只有再次在较小的球体内啄东西时才能得到食物。一直重复下去，直到鸽子可以在确定的位置上啄东西。

塑造法或自我塑造法可以改变单个动作的举动。但是一些动作顺序执行的动作链描述会更好。为了训练动物，实验人员利用高级条件反射（或条件强化）：就像经典条件反射中的那样，中性刺激可以增强与其配对的强化物的强化属性。而在反向链中，动物首先被训练来执行链中的最后一个动作，然后是倒数第二个，之后，它就处于可以执行之前动作的状态。在整个过程中，动物所处的不断变化的状态对以后的学习起到强化作用，直到整个序列都学会。

16.3.2 理论

为了解释试验中观察到的动作，Thorndike 提出了著名的效果律：

对同一刺激做出的响应，那些使动物获得满意的响应，会加强与这种刺激的联系。当再次出现这种刺激时，这种响应更有可能出现。但是那些使动物感到不适的响应，会减弱与这种刺激的联系。如果刺激再次发生，这种响应不太可能发生。满意或不适感越大，联系强化或者弱化的效果就越大 [Thorndike, 1911, Chapter 5, page 244]。

514

只有在实际执行了动作并对结果进行了评估，学习才会产生。根据 Thorndike 的理论，响应（或动作）与执行情况（状态）联系的强度决定了动作的强度。在许多 RL 算法中也使用了“效果律”中的基本概念。需要注意的是，动作的强度类似于动作的值 $Q(s, a)$ （其根据动作的结果而变化并且在许多 RL 方案中用于选择动作）。

在当前状态执行的动作有时候会被认为与刺激 - 反应（SR）之间的联系有关。尽管这个术语颇具争议，但是还是使用了它，因为在许多神经科学的动作理论中都用了它，并且它强调动作是由动作与当前状态之间的联系产生的。这种情况下，可以根据明确的预测，选择动作。

Thorndike 试图通过实验回答达尔文进化论中出现的问题：人类和动物有类似的心理能力吗？他的“效果律”对此做出了解释，如果再结合可变性（探索），甚至可以解释动物的一些复杂的动作。SR 的联系虽然简单，但在早期的行为心理学理论中起着重要作用例如，[Thorndike 1911 ; Hull 1943 ; Watson 1914]。新的 SR 的联系可以由之前学到的动作产生，并且复杂的响应可以由以前学习的简单响应组成。（这种概念也可以用于层次化的方法 [Grupen and Huber, 2005 ; Barto and Mahadevan, 2003]，见第 9 章）这个观点与进化论中的观点是一致的：动物也有能力理解人类的一些简单的动作的。（在他的书的第 6 章，Thorndike 认为思想和理性是人类的基本品质，源于我们高级的学习能力。）

与动物一样，人工智能学习特定的动作可能也是非常困难的。实验人员开发的训练程序，如反向链和塑造法，可用于辅助人工智能体去学习 [Konidaris and Barto, 2009 ;

Selfridge et al, 1985 ; Ng et al, 1999]。当使用强化方法来解决结构性奖励分配问题 [Barto, 1985] (当一个“动作”是由多个元素组成的, 一个学习器怎样选择元素才能获得奖励?) 时, 也会使用到塑造法。

与 SR 联系相类似的心理学理论提出了一种观点: 行为可以仅由直接观察的变量 (例如情境和反应) 来解释。这种观点有点极端, 不能解释所有的行为。下一小节讨论的实验表明, 一些无法直接观察到的变量却可以更好地解释这些行为。

16.3.3 基于模型的控制与无模型的控制

从 SR 的联系中获得动作的想法是基于动作的无模型解释——该动作的执行对结果的精确预测不起任何作用。但是, 一些实验的结果也提出了另一个理论, 从结果的精确预测中获得的动作是基于模型的解释, 对结果的预测起到很大作用。例如, 即使在没有任何奖励的情况下, 先前接触过迷宫的老鼠也会很快学会奖励的路线, 并一直会更快获得奖励。这种动作和其他迷宫实验的结果使得美国心理学家爱德华·托尔曼猜测: 老鼠会对环境建模, 并记录了状态之间以及动作与结果之间的关系 [Tolman, 1948]。在无模型的解释中, SR 的联系得到了强化, 而在基于模型的解释中, 动作与预测结果之间的联系得到了加强。

为了更直接地确定结果是否会对动作有影响, 操作人员设计了目标贬值 (或重估) 过程 [Dickinson, 1985 ; Dickinson and Balleine, 1994 ; Balleine et al, 2009]。在一个典型反射实验中, 训练一只动物来执行一些任务, 然后在另一种情况下, 改变它的奖励机制 (例如, 吃过食物会生病)。然后将动物放回到原来环境中使学习效果慢慢消退。(为了防止新的学习, 对消退结果不进行测试。) 如果动物的动作与没有经历目标贬值过程的动物的动作不同 (例如, 如果学习效果消退期间动作强度以更快的速度下降), 我们可以推断, 学习结果会对动物的动作产生一定的影响。

对于没有进行大量训练的动物, 情况确实如此, 表明它们确实使用的是基于模型的机制 (在许多情况下, 即使在贬值后的初次试验中, 动物也不执行有奖励的动作)。然而, 经过大量培训的动物的动作则不受贬值操作的影响。这种现象通常认为动物形成了习惯, 可以使用无模型的机制来解释。[Dickinson, 1985] 指出, 动作都是下面两个过程之一的结果: 随着动物不断地学习, 学习的速度和强化的速度之间存在很高的相关性, 这种相关性强化了动作-结果 (AO) 之间的关联。同时, 在这个过程中动物也在学习 SR 的联系。然而, 经过大量的训练之后, 学习的速度开始平稳, 与强化的速度之间的联系也变弱。因此, AO 关联减少, SR 的联系开始主导控制。

多种控制机制的可用性在模拟动物学习上具有功能优势。实验结果表明, 基于模型的控制器不需要很多经验, 这样在遇到奖励之后可以做出适当的决策。因此, 如果奖励结构发生了变化了, 它可以迅速适应, 甚至是立即适应。然而, 依据模型的预测所做出的决策, 对计算和表达性 (representational) 要求很高。无模型控制器需要较少的资源, 但需要较多的经验才能形成 SR 联系, 做出适当的决策。而这些要求使得它不那么灵活。

为了进一步了解学习过程, 我们可以使用含有多个控制器的计算模型。尤其, 在一个目标贬值任务中 [Daw et al, 2005], 基于模型和无模型的 RL 算法都可用来选择动作。由于任务在目标贬值时发生了变化, 因此不确定性也要纳入算法中。在每个状态下, 控制器都会建议执行不确定性较低的动作。因为无模型控制器需要较多的经验, 所以在前期, 基于模型的控制器起着主导作用。由于基于模型的控制器需要多个预测才能选择动作, 因此它能实现的

最小不确定性比无模型控制器的要高，无模型控制器主要在后期占主导地位。多控制器模型考虑了试验中观察到的大多数动作，并为这种动作做出了基于计算的解释。其他类型的多控制器模型提出了一种比较被动的判断方案：简单的控制器只要进行充分的训练就可以以较快的速度选择一个动作 [Ashby et al, 2007 ; Shah and Barto, 2009 ; Shah, 2008]。在构建层次化动作中，如技能或动作块可能会用到更简单的机制 [Shah, 2008]。

目标贬值实验中动物的动作表明动物在学习的不同阶段使用不同的控制机制。实验结果也表明，不同的脑系统参与调节不同的控制机制 [Balleine et al, 2009]。受这些结果的启发所构建的计算模型也使用了相应脑区的学习和控制机制。这些将在 16.5.3 节进一步讨论。

16.3.4 小结和其他注意事项

操作性条件反射实验表明，学会的动作可以用于训练动物学习其他具体的动作和调整该动作的强度（包括执行的可能性）。为了区别巴普洛夫动作，我们把这种动作称为操作性动作。心理学理论与 RL 有很多的共同点：标量增强信号强化了刺激和反应（SR，与无模型机制一致）或动作和结果（AO，基于模型的机制）之间的联系。

读者可能感兴趣的是动物行为实验中发现的其他几种过程。如，动物的动机状态会影响到它的动作。为了解释这个机制，赫尔提出了驱动理论：当动物的动力增加时，SR 的联系也会得到加强 [Hull, 1943]。而托尔曼的激励价值理论则认为：动作的强度取决于动物学习动作时所处的动机状态，与动力无关 [Tolman, 1949]。这些想法都经过了实验 [Dickinson and Balleine, 1994 ; Pompilio and Kacelnik, 2005] 和 RL 框架 [Niv et al, 2006b] 的验证。 [517]

在本节提到的实验中，当一个动作执行后，就会给一个奖励。但是其他实验只会在刺激导致了厌恶的结果（如电击）或者没有产生明显结果的时候检查动物的动作。其他类型的训练包括仅在执行了一些动作之后才会给奖励或者与上次奖励要有一段时间。这些操作导致了一些有趣的动作（如，匹配规则 [Herrnstein, 1961]），并可以进一步解释动物学习和控制的基本过程 [Staddon and Cerutti, 2003]。

一般来说，我们可以把动物当作 RL 学习器：修改它执行的动作，然后根据结果的反馈进行调节以得到最大的奖励。虽然我们可以通过研究动物的行为来了解更多机制。但是为了更好地了解这种行为背后的神经过程，我们需要了解相关的神经科学的研究。接下来的两节内容无不在说明：动物使用的学习方法与 RL 算法中使用的机制十分相似。

16.4 多巴胺

虽然不同的奖励机制对动物行为的影响是相似的，但是是否依靠不同的神经机制调节这些影响却是不知道的。在 20 世纪 50 年代初，研究发现如果一个动作导致传递给大脑一个电刺激信号，动作的强度就会增加 [Olds and Milner, 1954]。当电极强烈刺激神经突触导致神经末梢释放多巴胺时，这种颅内自我刺激（ICSS）作用会变得更加明显。DA 神经元大多位于黑质致密部（SNPc，也称为 A9 组）和腹侧被盖区附近（VTA，A10 组）[Björklund and Dunnett, 2007]。如下一节所述，DA 神经元涉及决策和动作的许多领域。ICSS 表明，动作可能会根据 DA 神经元传达的信号而发生改变。最初的解释是，DA 直接预测奖励的发生。随后的研究表明 DA 可能发挥着更加复杂的作用 [Wise, 2004 ; Schultz, 2007 ; Bromberg-Martin et al, 2010 ; Montague et al, 2004]。

16.4.1 多巴胺作为奖励预测误差

为了更好地研究 DA 在动作中的作用, 研究人员记录了 VTA 和 SNpc 中单个神经元在动物学习中的活动。DA 神经元活动表现出较低的基线初始值 (小于 10 赫兹) 和短暂的放电 (相位) (以下称为“DA 爆发”, 爆发的强度是指爆发中发射的频率)。早期的研究表明, 在响应与任务相关的、激烈或者惊奇的事件时, 会出现 DA 爆发 [Miller et al, 1981; Schultz, 1986; Horvitz, 2000]。

在探究 RL 与脑活动之间关系的研究中, [Ljungberg et al, 1992] 记录了猴子在学习光出现的时候伸手去拿一个杠杆, 然后它们会得到一些果汁这个过程中 AD 神经元的活动。最初, DA 爆发在果汁交付的时候产生。随着不断地学习, DA 爆发在出现光和交付果汁的时候都会产生, 后来只会在发现光的时候产生。最后, 经过大约 30 000 次试验, 即使在出现光的时候, DA 爆发产生的次数也比较少。同样地, [Schultz et al, 1993] 也表明, 随着猴子们学习更复杂的操作性条件反射任务, DA 爆发发生的时间从果汁交付时刻转换到了刺激产生的时刻。如果猴子执行了错误的动作, 则在预期果汁交付的时刻, DA 神经元活动会降低。此外, 如果在预期的时间之前就交付果汁, 则会导致 DA 爆发; 如果在预期的时间不递交果汁 (即使猴子动作正确) 则会导致 DA 神经元活动减少; 随着猴子不断地学习, 即使在交付果汁时刻 DA 爆发产生的次数也会减少 [Schultz et al, 1997; Hollerman and Schultz, 1998]。

在学习过程中, DA 神经元的活动与在实验中直接设定的变量没有关系, 这引起了一些人的注意。如 [Houket et al, 1995] 所述, “DA 神经元的放电特性与 TD 算法产生的有效强化信号之间存在明显的相似性”。[Montague et al, 1996] 猜测“从 VTA 向皮质和皮层下靶结构传递的多巴胺的量在一定程度上表明了预期奖励量与实际奖励量之间误差的信息”。[Schultz et al, 1997] 更详细地讨论了他们的实验, TD 和心理学习理论之间的关系。这些相似之处的重要性不言而喻: 在 RL 框架内形成的一个基本的学习信号——几乎完全可以表示 DA 神经元的活动。

一些研究表明, DA 爆发与 TD 误差一样受到了刺激的预测特性的影响, 例如阻塞 [Waelti et al, 2001] 和条件抑制 [Tobler et al, 2003]。训练过的猴子受到刺激 (能预测到有一定概率会得到奖励) 时, DA 爆发的强度会随着得到奖励的概率的增加而增加; 随着奖励出现的概率的减少而减少。而当没有奖励的时候, DA 神经元的活跃度则会大幅度的减少 [Fiorillo et al, 2003]。当奖励的大小由概率分布决定时, 奖励交付时 DA 爆发的多少反映了实际奖励与期望奖励的大小之间的差异 [Tobler et al, 2005]。影响 DA 爆发的其他因素包括动机 [Satoh et al, 2003]、奖励递交延迟 [Kobayashi and Schultz, 2008] 和历史 (如果奖励交付是过去事件的函数) [Nakahara et al, 2004; Bayer and Glimcher, 2005]。

为了研究 DA 神经元活动在决策过程中 [Morris et al, 2006] 的作用, 我们给猴子一个视觉刺激 (可以预测得到奖励的概率)。如果仅有一个刺激, 则刺激呈现时的 DA 神经元活动与预期值成比例。如果有两种刺激, 并且猴子可以在它们之间进行选择时, 则 DA 神经元活动与最终选择的刺激的预期值成比例, 而不是可选择的刺激期望值的某种组合。笔者认为, 这些结果表明, DA 神经元活动反映了对所选择动作的认知价值, 与 SARSA 学习方案 [Niv et al, 2006a] 一致 (尽管其他实验的结果支持 Q 学习方案 [Roesch et al, 2007])。

16.4.2 多巴胺的强化信号的作用

上述研究也表明了: DA 爆发所反映的奖励预测误差与 RL 中的 TD 误差非常的相似。

这个结论非常具有吸引力，因为我们可以利用 RL 中丰富的计算框架来分析这种活动。然而，其他研究和解释表明：DA 只是作为动作的一般强化物，也许不仅仅与奖励最大化有关。

激励突出理论 [Berridge and Robinson, 1998; Berridge, 2007; Berridge et al, 2009] 将“欲望”（动作偏好）与“喜欢”（享乐感）区分开来。与药理学操作相关的实验表明：阿片类物质系统调节着与快乐相关的面部表情 [Berridge and Robinson, 1998]。当上述两个没有区分开时，可以解释为什么在没有增加操作时 [Tindell et al, 2005; Wyvell and Berridge, 2000]，DA 可以增加动作的强度和 DA 缺陷的动物更偏好学习中性刺激 [Cannon and Palmiter, 2003]。

但是将这两个分离开来也为一些实验结果和非理性动作做出了合理的解释 [Wyvell and Berridge, 2000]。

[Redgrave et al, 2008] 认为，DA 爆发的延迟（在许多情况下，是刺激出现在 100ms 以内）太短并且是均匀的，导致无法识别和预测刺激。相反，DA 爆发可能是由于完全皮质下通路的响应较快（比 DA 神经元响应更快），预先粗略感知到某些事情会发生，但不知道它是什么 [Dommett et al, 2005]。最近的研究结果表明：DA 神经元的活动出现了额外的阶段（较长的延迟（ $<200\text{ms}$ ）），这可能是由早期皮层处理和提供了与奖励相关的信息造成的。早期的（ $<100\text{ms}$ ）DA 活动可能表示感官预测误差 [Horvitz, 2000]（偏向于让动物重复选择动作，以确定它做了什么），如果存在 DA 活动，则表示引起了感官事件。后期的 DA 活动可能还会使动物重复某个动作以使得奖励最大化。 [520]

16.4.3 小结和其他注意事项

本节的实验结果表明，DA 不是“奖励探索器”而是强化动作的学习信号。通过药物来控制 DA 的有效性，进一步支持这一观点。例如，在执行任务时给予 DA 激动剂的人（增加 DA 在靶神经元上的有效性），在纹状体（DA 神经元靶向的脑区）中与学习和 TD 误差有关的活动都有所增加。DA 拮抗剂（降低 DA 的有效性）具有相反的作用。

有一些有趣的问题，我没有讨论，但值得一提。比如说，“DA 爆发是如何形成的”是一个值得讨论的问题。一些理论（基于对 DA 神经元的预测）认为，当预测奖励发生时，DA 活动会慢慢被抑制 [Hazy et al, 2010; Houk et al, 1995]。此外，由于 DA 神经元活动基线较低，产生厌恶结果或取消奖励不能以与交付奖励相同的方式来表示。基于激励的表示理论 [Ludvig et al, 2008, Daw et al, 2006a] 其他神经递质系统 [Daw et al, 2002; Phelps and LeDoux, 2005; Wrase et al, 2007; Doya, 2008] 和 / 或学习规则 [Frank, 2005; Frank et al, 2004] 解决了这个问题。虽然本节重点是介绍 DA 神经元活动，但是也讨论了长期的 DA 神经元活动对学习和行为的影响 [Schultz, 2007; Daw and Touretzky, 2002]。最后，最近的实验表明，DA 神经元的作用在解剖学的解释可能与本节所述的有所出入 [Haber, 2003; Wickens et al, 2007]。

虽然关于 DA 如何影响动作和 DA 最终影响动作的是什么已经提出了很多的假设，但本章令读者最感兴趣的应该是 DA 爆发与 TD 误差所表示的信号居然非常相似。为了确定该信号在大脑中的运行的方法是否与 RL 算法的方法相似，我们必须检查 DA 神经元的靶结构。

16.5 基底神经节

大多数 DA 神经元投影于额叶皮层和基底神经节（BG），大脑中控制动作、决策和其他

认知过程的区域。因为 DA 到 BG 的投影比到额叶皮层的投影多，所以本节重点放在了 BG 上，以下是对 BG 及其在学习和控制中作用的简要概述。

16.5.1 基底神经节概述

BG 是位于丘脑附近的一组相互连接的皮质下结构。二十世纪初，科学家们首先将它们与自愿运动联系起来。当时验尸分析显示，在帕金森病患者脑中一部分 BG 受损。随后的研究发现它们也与运动有关 [Mink, 1996]，并且过去几十年的研究发现，它们也调节着学习和认知功能 [Packard and Knowlton, 2002; Graybiel, 2005]。

BG 中的一部分称为纹状体，它不仅接受来自 DA 神经元的投影而且还能够接受来自多数皮层和丘脑区域的兴奋性投影。纹状体神经元会从皮层神经元接收大量弱输入，表明纹状体神经元可以进行模式识别 [Houk and Wise, 1995; Wilson, 2004]。纹状体神经元向苍白球 (GPi) 内核发出抑制性投影，但是苍白球内核的神经元是调节活性的并且会向脑干和丘脑发出抑制性投影。因此，在纹状体神经元的激发下，会导致 GPi 神经元的靶向神经元的抑制得到解除。当动作导致了 GPi 靶向神经元激活时，抑制解除增加了动作被执行的可能性。

在抽象层面上，纹状体神经元中的模式识别与 RL 中所使用的状态检测十分相类似，并且 GPi 靶向神经元的去抑制结果与动作选择结果很类似。皮质纹状体突触会受到 DA 可塑性影响 [Wickens, 2009; Calabresi et al, 2007]。例如，一个学习规则大致是纹状体神经元的活动以及皮质投影和 DA 神经元活动的产物。因此，DA 爆发（例如，表示 TD 误差）可以改变纹状体神经元的活动，使得它可以根据动作的结果对特定状态做出回应。换句话说，BG 能够使用与 RL 中类似的机制来改变动作的选择 [Barto, 1995; Doya, 1999; Daw and Doya, 2006; Doll and Frank, 2009; Graybiel, 2005; Joel et al, 2002; Wickens et al, 2007]。

BG 内的其他路径（这里不再描述）赋予 BG 控制动作的能力。例如，可以通过不同的学习机制来促进或抑制动作的执行 [Frank, 2005; Frank et al, 2004; Hikosaka, 2007]。而且，BG 内部架构似乎非常适合在具有相互竞争的动作之间进行选择 [Bogacz and Gurney, 2007; Gurney et al, 2001]。

皮层的不同区域（涉及不同类型的功能）会投影到纹状体的不同区域。这些投影回路在很大程度上是通过 BG 到丘脑，再回到大脑皮层的 [Alexander et al, 1986; Middleton and Strick, 2002]。这种并行循环结构允许 BG 通过形成皮质活动和降低向脑干的投影来影响动作。并行循环结构的功能将在稍后讨论。下一节将进一步阐述 BG 的功能，主要侧重于动作和决策选择方面。

16.5.2 纹状体的神经活动

在本节描述的大多数实验中，当动物参与了一些条件反射实验时，我们会记录纹状体中单个神经元的活动。研究结果表明随着动物的不断学习，神经活动（包括奖励所调节的活动）开始与任务相关联 [Schultz et al, 2003; Hikosaka, 2007; Barnes et al, 2005]。

在一项与此相关的特别的研究中，[Samejima et al, 2005] 记录了参与任务（在两个动作中自由选择一个）的猴子脑中背侧纹状体的变化。在实验中，每个动作都有一定概率得到一个大的奖励（一大杯果汁）或一个小的奖励（少量果汁）。例如，在一个实验中，动作 A 导

致大奖励的概率为 0.5，动作 B 的概率为 0.9，而在另一个实验中，A 和 B 获得大奖励的概率分别为 0.5 和 0.1。这样的设计使得动作的绝对值和相对值分离开来（动作的值表示执行动作后，动物可获得奖励的概率）：动作 A 的值在两个实验中都相同，但是低于第一个实验中动作 B 的值，高于第二个实验中 B 的值。在试验中，动物往往会优先选择值较高的动作。

在实验中，记录中大约三分之一的神经元的活动都与动作的值有关（而其他方面，如动作值的差异或最终的选择则占比较少的比例）。此外，我们可以使用建模技术根据经验（即过去的动作和奖励）实时估计动作的值，并根据估计出的动作值来预测动物选择的动作。实验结果发现，许多神经元的活动都与估计的动作值有关，并且预测的选择分布与实际观察到的分布相吻合。

通过一些实验，我们会进一步加深对神经活动的时间特征的理解。[Lau and Glimcher, 2008] 的实验表明：对可选择的动作的值进行编码的背侧纹状体神经元在动作执行之前会变得更加活跃。而对选择过的动作的值进行编码的那些背侧纹状体神经元则会在动作执行之后变得更加活跃。这些和一些其他的研究结果 [Balleine et al, 2007 ; Kim et al, 2009] 表明，某些形式的动作的值存在于背侧纹状体结构中，并且参与了动作的选择和评估。（尽管一些病变研究表明，背侧纹状体可能不需要知道这些值 [Atallah et al, 2007]）。

[Samejima et al, 2005] 的分析指出了一种在过去十几年中越来越重视的方法：不仅要神经活动与可直接观察的变量（例如奖励或选择）相关联，而且还要考虑理论和计算模型中参与学习和控制的变量（例如，期望值）[Corrado and Doya, 2007 ; Daw and Doya, 2006 ; Niv, 2009]。这种方法在分析从功能性核磁共振成像（fMRI）方法 [Gläscher and O'Doherty, 2010 ; Montague et al, 2006 ; Haruno and Kawato, 2006] 中获得的数据时特别有用，并且可以同时记录许多脑区的活动，甚至可以记录从事复杂认知任务人脑区的活动。值得注意的是，被测信号（血液中含氧量）和神经活动之间的精确关系是未知的并且信号与单个神经元记录的关系无论是在时间上还是空间上都比较难测量。话虽如此，但是大量的数据分析却可以让我们更好地了解脑区之间的相互作用。

[Doherty et al, 2004] 使用 fMRI 方法发现：在人类参与者必须从一组动作中选出一个动作的试验中，人类的背侧纹状体和腹侧纹状体都出现了像 TD 误差一样的信号。而当参与者不需要做出选择时，只有腹侧纹状体出现了这样的信号。这些结果表明背侧和腹侧纹状体分别实现了类似于 Actor 和 Critic，参见 [Barto 1995, Joel et al 2002, Montague et al 2006] 的功能。

来自动物腹侧纹状体中单个神经元的记录分析结果肯定了这一观点。用于评估动作的信息，如语境（或状态）、某些动作和结果，似乎都存在于腹侧纹状体中 [Ito and Doya, 2009 ; Roesch et al, 2009 ; Kim et al, 2009]。简单来说就是，背侧纹状体更多的是控制一些一般的动作，而腹侧纹状体可能参与赋予刺激的值，但也可能参与了某些动作的控制，并在其中发挥着更为复杂的作用 [Yin et al, 2008 ; Humphries and Prescott, 2010 ; Nicola, 2007]。

16.5.3 皮质基神经节丘脑循环

如前所述，从皮层到 BG 再到丘脑，最后再回到皮层的投影路径形成了平行隔离环 [Alexander et al, 1986 ; Middleton and Strick, 2002]。环内 BG 的基本作用与环内神经活动所表示的信息有关 [Wickens et al, 2007 ; Samejima and Doya, 2007 ; Graybiel et al, 1994 ; Houk et al, 2007 ; Yin et al, 2008 ; Haruno and Kawato, 2006 ; Redgrave et al, 2010 ;

523

Packard and Knowlton, 2002; Cohen and Frank, 2009] 此外, 由于不同皮质区域涉及不同类型的功能, 并行循环结构允许 BG 通过不同的机制控制动作 (例如, 那些在经典和操作性条件反射控制的动作, 参见 16.2.4 节和 16.3.3 节)。还有一些解释与 [Yin et al, 2008] 提出理论的很类似: 1) 通过无模型机制学习操作性动作 (即由刺激反应产生, SR, 关联); 2) 通过基于模型的机制学习的操作性动作 (动作结果, AO, 关联); 3) 学习刺激结果, SO, 关联, 这可能导致巴普洛夫动作。

524 无模型机制通常被认为是在背外侧纹状体 (DLS, 在灵长类动物中也叫壳核) 的环内实现的。DLS 主要从初级感觉、运动神经和前运动皮质 (其统称为感觉运动皮质或 SMC) 中接受皮质投影, 它们也为 DLS 提供了基本的感知和动作信息。

基于模型的机制被认为是在背内侧纹状体 (DMS, 也称尾状物) 的环内实现的, 其主要接受来自前额叶皮层 (PFC) 的皮质投影。PFC 位于皮质的前部, 与许多其他皮质区域相互连接, 控制着感觉和动作。PFC 神经元可保持持续的活动 (工作记忆 [Goldman-Rakic, 1995]), 因此可以使用它们临时存储信息。由 DMS 调节的 RL 机制决定了过去的哪些刺激可由持续活动的神经元表示 [O'Reilly and Frank, 2006]。此外, PFC 通常被认为参与了环境模型的构建 [Gläscher et al, 2010], 从而能够存储预测到的未来事件。因此, PFC 可以通过制定规划来影响动作, 影响力甚至超越了其他脑区 [Miller and Cohen, 2001; Tanji and Hoshi, 2008]。

中性刺激值的分配是在腹侧纹状体 (VS, 也称为伏隔核) 的环路中实现, 该环路主要接受来自眶额叶皮层 (OFC, 刚好位于额头后面的 PFC 的下面部分) 的皮质投射。VS 和 OFC 都与边缘区域有联系, 如杏仁核和下丘脑。这些结构和 VS、OFC 都与情绪、动作、奖励有关 [Wallis, 2007; Cardinal et al, 2002; Mirolli et al, 2010]。

侧重于循环结构的大多数脑功能理论都用到了 [Yin et al, 2008] 提出的理论 (尽管有一些差异) [Samejima and Doya, 2007; Haruno and Kawato, 2006; Daw et al, 2005; Balleine et al, 2009; Ashby et al, 2010; Balleine and O'Doherty, 2010; Pennartz et al, 2009; Houk et al, 2007; Wickens et al, 2007, Redgrave et al, 2010; Mirolli et al, 2010; Packard and Knowlton, 2002; Cohen and Frank, 2009]。所有人的共同观点是, 不同的循环实现了不同的学习与控制机制。对动作的研究表明不同机制主导控制着学习中的不同点 (例如, 控制可以从基于模型的机制转换到无模型的机制, 如 16.3.3 节所述), 神经科学研究表明, 与循环相关的脑结构主导控制着学习的不同点 [Doyon et al 2009; Poldrack et al 2005]。

例如, 当人学习一些动作时, PFC (基于模型的机制) 在学习前期支配着大脑活动 (通过 fMRI 测量), 而纹状体和 SMC (无模型机制) 则在学习的后期主导大脑活动 [Doyon et al, 2009]。这些结果表明在基于模型的控制中, 决策机制主要位于 PFC 内, 而在无模型控制期间, 决策机制主要位于皮质纹状体到 DLS 的突触上。也就是说, 控制从皮层选择机制转换到了 BG 选择机制 [Daw et al, 2005; Niv et al, 2006b; Shah, 2008; Shah and Barto, 2009], 这与实验研究结果一致, 因此也说明 BG 在对运动技能的编码和习惯动作方面发挥重要作用 [Graybiel, 2008; Pennartz et al, 2009; Aldridge and Berridge, 1998]。然而, 其他理论和实验结果却发现控制转换方向是相反的: BG 参与早期的试验和试误学习

525 [Pasupathy and Miller, 2005; Packard and Knowlton, 2002], 而皮质区域参与调节习惯或技术动作 [Ashby et al, 2007, 2010; Frank and Claus, 2006; Matsuzaka et al, 2007]。如 [Ashby et al, 2010] 所讨论的, 这些差异可能是由于使用了不同的实验方法 (包括动物执行的任务

和用于定义习惯或熟练动作的措施)。

最后,一些研究重点是如何通过不同的环路控制动作。由一个循环中的机制产生的动作可以用于训练另一个循环的机制,但不同循环之间会有一些通信 [Haber, 2003; Haber et al, 2006; Pennartz et al, 2009; Yin et al, 2008; Balleine and O'Doherty, 2010; Mirolli et al, 2010; Joel and Weiner, 1994; Graybiel et al, 1994]。这种通信可以发生在 BG 内 [Pennartz et al, 2009; Graybiel, 2008; Joel and Weiner, 1994; Graybiel et al, 1994], 或通过纹状体和 DA 神经元之间的连接产生 (其中一些也是 BG 的一部分)。后一种通信过程是这样的: 纹状体的一个区域会投射到神经元上, 这些神经元也会向它发送投影, 有些还会传送到纹状体的一个相邻区域 [Haber, 2003]。连通性模式形成了一个螺旋, 在螺旋中通信主要沿一个方向传递, 并形成了一个层次化的组织, 与较高级别循环相关联的学习可以用于训练较低级别循环中的学习 [Haruno and Kawato, 2006; Yin et al, 2008; Samejima and Doya, 2007]。例如类似于 [Yin et al, 2008], OFC-VS 环路 (SO 关联) 可以训练 SMC-DLS 环路 (SR), 而 SR 可以训练 PFC-DMS 环路 (AO)。

16.5.4 小结和其他注意事项

BG 功能受 DA 影响很大, 我们可以将其近似表示为 TD 误差。BG 的结构和生理特性赋予了它 RL 的能力。背侧纹状体的神经元 (那些与大脑中控制着动作和决策相关的区域) 决定着动作的值。这些结果表明, BG 以类似于 RL 算法中所使用的方式调节和控制着学习。

研究还表明, 不同皮层-基底神经节-丘脑循环使用不同类型信息的 RL 机制。并行环路结构允许使用不同的机制来影响动作, 循环之间的通信可以使得在一种机制中的学习驱动另一种机制中的学习。衍生自生物系统的概念框架表明为人工智能构建层次化的控制框架 (包括不同层次间的不同信息的表示, 以及不同级别间如何通信) 具有非常大的潜在优势, 这种优势不仅体现在不同层次的结构代表什么类型的信息, 还表现在不同层次间的相互通信上。

526

当然, 除了本节所述的因素之外, 路径、脑区和功能机制也影响学习和控制。例如, DA 直接影响神经活动, 纹状体活动则是由中间神经元 (投射到同一结构内的其他神经元) 的活动形成的, 这些活动都随着动物的学习而发生巨大变化 [Graybiel et al, 1994; Graybiel, 2008; Pennartz et al, 2009], 并且 BG 还可以通过与皮层下结构进行复发性连接来影响动作 [McHaffie et al, 2005]。在额叶皮质区中与奖励相关的神经活动 [Schultz, 2006] 是通过 DA 的直接投影、其他脑区的交互以及与 BG 交互形成的。

这里所没有讨论但通常用于 RL 和机器学习的计算机制和思路, 在大脑中也具有类似的东西。心理和神经科学的研究涉及诸如不确定性动作、状态抽象、博弈理论、探索与开发、层次化动作之类的课题 [Dayan and Daw, 2008; Doya, 2008; Gold and Shadlen, 2007; Seger and Miller, 2010; Glimcher and Rustichini, 2004; Daw et al, 2006b; Yu and Dayan, 2005; Wolpert, 2007; Botvinick et al, 2009; Grafton and Hamilton, 2007]。准确确定大脑区域是如何作用于动作是非常困难的。使用前面讨论的方法 (根据理论计算解释中使用的变量分析大脑活动 [Corrado and Doya, 2007; Daw and Doya, 2006; Niv, 2009]) 来分析脑中各个区域已经超出了本节所讨论的范围。例如, 神经经济学领域是研究人类在参与经济博弈时, 大脑中的决策过程以及相关的脑区的活动的 [Glimcher and Rustichini, 2004; Glimcher, 2003]。

16.6 总结

RL 是学习的一个计算公式,它通过与环境进行交互,选择执行一些可以得到满意结果的动作。动物天生拥有这种能力。并且 RL 在很大程度上是受到早期动物行为学和心理学理论的启发形成的(见 16.2 节和 16.3 节)。生物学的进步使得科学家能够通过记录动物在学习任务时的神经活动,从而更好地理解动物的学习和控制机制。RL 提供了一个框架来分析这种活动,并且发现大脑使用的一些机制与 RL 算法中使用的机制非常相似(见 16.4 节和 16.5 节)。

尤其,根据 TD 误差(见 16.4.1 节)对 DA 神经元活动的表征,可以在 RL(和机器学习)与心理学和神经科学之间建立更强且清晰的通信。已出版了大量关注这些连接的优秀论文 [Niv 2009; Dayan and Daw 2008; Daw and Doya 2006; Cohen 2008; Doya 2007; Maia 2009; Dayan and Niv 2008]。此外,计算神经科学研究解释了大脑区域的生理和结构特征是如何实现特定功能的 [Wörgötter and Porr 2005; Cohen and Frank 2009; Gurney et al 2004; Gurney 2009; Doya 2008; Bar-Gad et al 2003; Joel et al 2002; Hazy et al 2010; Mirolli et al 2010]。计算方法与心理学和神经科学的结合不仅使我们更好地了解决策过程,而且也为我们研究决策障碍(精神障碍和成瘾)提供了新的方法 [Maia and Frank, 2011; Kishida et al, 2010; Redish et al, 2008; Redgrave et al, 2010; Graybiel, 2008; Belin et al, 2009]。

最后,最近神经科学的研究开始进一步揭开动物复杂动作的机制。尤其,动物似乎是使用多个控制机制而不是使用单一的控制机制来控制动作的(见 16.3.3 节和 16.5.3 节)。神经科学研究发现了这些不同的机制间是如何相互协作的(见 16.5.3 节)。这些发现不仅为研究动物的复杂动作做出了贡献,并且为构建复杂的自动人工智能提供了灵感。

致谢。我很感激来自 Andrew Barto、Tom Stafford、Kevin Gurney、Peter Redgrav 以及匿名评论家的评论,还有来自欧洲共同体第七框架计划 231722(IM-CLeVeR)提供的财政支持。

参考文献

- Aldridge, J.W., Berridge, K.C.: Coding of serial order by neostriatal neurons: a “natural action” approach to movement sequence. *The Journal of Neuroscience* 18, 2777–2787 (1998)
- Alexander, G.E., DeLong, M.R., Strick, P.L.: Parallel organization of functionally segregated circuits linking basal ganglia and cortex. *Annual Review of Neuroscience* 9, 357–381 (1986)
- Ashby, F.G., Ennis, J., Spiering, B.: A neurobiological theory of automaticity in perceptual categorization. *Psychological Review* 114, 632–656 (2007)
- Ashby, F.G., Turner, B.O., Horvitz, J.C.: Cortical and basal ganglia contributions to habit learning and automaticity. *Trends in Cognitive Sciences* 14, 208–215 (2010)
- Atallah, H.E., Lopez-Paniagua, D., Rudy, J.W., O’Reilly, R.C.: Separate neural substrates for skill learning and performance in ventral and dorsal striatum. *Nature Neuroscience* 10, 126–131 (2007)
- Balleine, B.W., O’Doherty, J.P.: Human and rodent homologues in action control: Corticostriatal determinants of goal-directed and habitual action. *Neuropsychopharmacology* 35, 48–69 (2010)
- Balleine, B.W., Delgado, M.R., Hikosaka, O.: The role of the dorsal striatum in reward and decision-making. *The Journal of Neuroscience* 27, 8161–8165 (2007)
- Balleine, B.W., Liljeholm, M., Ostlund, S.B.: The integrative function of the basal ganglia in

- instrumental conditioning. *Behavioural Brain Research* 199, 43–52 (2009)
- Bar-Gad, I., Morris, G., Bergman, H.: Information processing, dimensionality reduction, and reinforcement learning in the basal ganglia. *Progress in Neurobiology* 71, 439–473 (2003)
- Barnes, T.D., Kubota, Y., Hu, D., Jin, D.Z., Graybiel, A.M.: Activity of striatal neurons reflects dynamic encoding and recoding of procedural memories. *Nature* 437, 1158–1161 (2005)
- Barto, A.G.: Learning by statistical cooperation of self-interested neuron-like computing elements. *Human Neurobiology* 4, 229–256 (1985)
- Barto, A.G.: Adaptive critics and the basal ganglia. In: Houk, J.C., Davis, J.L., Beiser, D.G. (eds.) *Models of Information Processing in the Basal Ganglia*, ch. 11, pp. 215–232. MIT Press, Cambridge (1995)
- Barto, A.G., Mahadevan, S.: Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems* 13, 341–379 (2003)
- Barto, A.G., Sutton, R.S.: Simulation of anticipatory responses in classical conditioning by a neuron-like adaptive element. *Behavioral Brain Research* 4, 221–235 (1982)
- Barto, A.G., Sutton, R.S., Anderson, C.W.: Neuronlike elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics* 13, 835–846 (1983)
- Bayer, H.M., Glimcher, P.W.: Midbrain dopamine neurons encode a quantitative reward prediction error signal. *Neuron* 47, 129–141 (2005)
- Belin, D., Jonkman, S., Dickinson, A., Robbins, T.W., Everitt, B.J.: Parallel and interactive learning processes within the basal ganglia: relevance for the understanding of addiction. *Behavioural Brain Research* 199, 89–102 (2009)
- Berridge, K.C.: The debate over dopamine's role in reward: The case for incentive salience. *Psychopharmacology* 191, 391–431 (2007)
- Berridge, K.C., Robinson, T.E.: What is the role of dopamine in reward: Hedonic impact, reward learning, or incentive salience? *Brain Research Reviews* 28, 309–369 (1998)
- Berridge, K.C., Robinson, T.E., Aldridge, J.W.: Dissecting components of reward: 'Liking,' 'wanting,' and learning. *Current Opinion in Pharmacology* 9, 65–73 (2009)
- Björklund, A., Dunnett, S.B.: Dopamine neuron systems in the brain: an update. *Trends in Neurosciences* 30, 194–202 (2007)
- Bogacz, R., Gurney, K.: The basal ganglia and cortex implement optimal decision making between alternative actions. *Neural Computation* 19, 442–477 (2007)
- Botvinick, M.M., Niv, Y., Barto, A.G.: Hierarchically organized behavior and its neural foundations: A reinforcement-learning perspective. *Cognition* 113, 262–280 (2009)
- Brandon, S.E., Vogel, E.G., Wagner, A.R.: Computational theories of classical conditioning. In: Moore, J.W. (ed.) *A Neuroscientist's Guide to Classical Conditioning*, ch. 7, pp. 232–310. Springer, New York (2002)
- Bromberg-Martin, E.S., Matsumoto, M., Hikosaka, O.: Dopamine in motivational control: Rewarding, aversive, and alerting. *Neuron* 68, 815–834 (2010)
- Brown, P.L., Jenkins, H.M.: Auto-shaping of the pigeon's key-peck. *Journal of the Experimental Analysis of Behavior* 11, 1–8 (1968)
- Calabresi, P., Picconi, B., Tozzi, A., DiFilippo, M.: Dopamine-mediated regulation of corticostriatal synaptic plasticity. *Trends in Neuroscience* 30, 211–219 (2007)
- Cannon, C.M., Palmiter, R.D.: Reward without dopamine. *Journal of Neuroscience* 23, 10,827–10,831 (2003)
- Cardinal, R.N., Parkinson, J.A., Hall, J., Everitt, B.J.: Emotion and motivation: The role of the amygdala, ventral striatum, and prefrontal cortex. *Neuroscience and Biobehavioural Reviews* 26, 321–352 (2002)
- Cohen, M.X.: Neurocomputational mechanisms of reinforcement-guided learning in humans: a review. *Cognitive, Affective, and Behavioral Neuroscience* 8, 113–125 (2008)
- Cohen, M.X., Frank, M.J.: Neurocomputational models of the basal ganglia in learning, memory, and choice. *Behavioural Brain Research* 199, 141–156 (2009)
- Corrado, G., Doya, K.: Understanding neural coding through the model-based analysis of decision-making. *The Journal of Neuroscience* 27, 8178–8180 (2007)

- Daw, N.D., Doya, K.: The computational neurobiology of learning and reward. *Current Opinion in Neurobiology* 16, 199–204 (2006)
- Daw, N.D., Touretzky, D.S.: Long-term reward prediction in TD models of the dopamine system. *Neural Computation* 14, 2567–2583 (2002)
- Daw, N.D., Kakade, S., Dayan, P.: Opponent interactions between serotonin and dopamine. *Neural Networks* 15, 603–616 (2002)
- Daw, N.D., Niv, Y., Dayan, P.: Uncertainty-based competition between prefrontal and dorso-lateral striatal systems for behavioral control. *Nature Neuroscience* 8, 1704–1711 (2005)
- Daw, N.D., Courville, A.C., Touretzky, D.S.: Representation and timing in theories of the dopamine system. *Neural Computation* 18, 1637–1677 (2006a)
- Daw, N.D., O'Doherty, J.P., Dayan, P., Seymour, B., Dolan, R.J.: Cortical substrates for exploratory decisions in humans. *Nature* 441, 876–879 (2006b)
- Dayan, P., Daw, N.D.: Connections between computational and neurobiological perspectives on decision making. *Cognitive, Affective, and Behavioral Neuroscience* 8, 429–453 (2008)
- Dayan, P., Niv, Y.: Reinforcement learning: the good, the bad, and the ugly. *Current Opinion in Neurobiology* 18, 185–196 (2008)
- Dayan, P., Niv, Y., Seymour, B., Daw, N.D.: The misbehavior of value and the discipline of the will. *Neural Networks* 19, 1153–1160 (2006)
- Dickinson, A.: Actions and habits: the development of behavioural autonomy. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 308, 67–78 (1985)
- Dickinson, A., Balleine, B.W.: Motivational control of goal-directed action. *Animal Learning and Behavior* 22, 1–18 (1994)
- Doll, B.B., Frank, M.J.: The basal ganglia in reward and decision making: computational models and empirical studies. In: Dreher, J., Tremblay, L. (eds.) *Handbook of Reward and Decision Making*, ch. 19, pp. 399–425. Academic Press, Oxford (2009)
- Dommett, E., Coizet, V., Blaha, C.D., Martindale, J., Lefebvre, V., Mayhew, N.W.J.E., Overton, P.G., Redgrave, P.: How visual stimuli activate dopaminergic neurons at short latency. *Science* 307, 1476–1479 (2005)
- Doya, K.: What are the computations of the cerebellum, the basal ganglia, and the cerebral cortex? *Neural Networks* 12, 961–974 (1999)
- Doya, K.: Reinforcement learning: Computational theory and biological mechanisms. *HFSP Journal* 1, 30–40 (2007)
- Doya, K.: Modulators of decision making. *Nature Neuroscience* 11, 410–416 (2008)
- Doyon, J., Bellec, P., Amsel, R., Penhune, V., Monchi, O., Carrier, J., Lehericy, S., Benali, H.: Contributions of the basal ganglia and functionally related brain structures to motor learning. *Behavioural Brain Research* 199, 61–75 (2009)
- Eckerman, D.A., Hienz, R.D., Stern, S., Kowlowitz, V.: Shaping the location of a pigeon's peck: Effect of rate and size of shaping steps. *Journal of the Experimental Analysis of Behavior* 33, 299–310 (1980)
- Ferster, C.B., Skinner, B.F.: *Schedules of Reinforcement*. Appleton-Century-Crofts, New York (1957)
- Fiorillo, C.D., Tobler, P.N., Schultz, W.: Discrete coding of reward probability and uncertainty by dopamine neurons. *Science* 299, 1898–1902 (2003)
- Frank, M.J.: Dynamic dopamine modulation in the basal ganglia: a neurocomputational account of cognitive deficits in medicated and nonmedicated Parkinsonism. *Journal of Cognitive Neuroscience* 17, 51–72 (2005)
- Frank, M.J., Claus, E.D.: Anatomy of a decision: Striato-orbitofrontal interactions in reinforcement learning, decision making, and reversal. *Psychological Review* 113, 300–326 (2006)
- Frank, M.J., Seeberger, L.C., O'Reilly, R.C.: By carrot or by stick: Cognitive reinforcement learning in parkinsonism. *Science* 306, 1940–1943 (2004)
- Gardner, R.: Multiple-choice decision behavior. *American Journal of Psychology* 71, 710–717 (1958)
- Gläscher, J.P., O'Doherty, J.P.: Model-based approaches to neuroimaging combining reinforcement learning theory with fMRI data. *Wiley Interdisciplinary Reviews: Cognitive*

- Science 1, 501–510 (2010)
- Gläscher, J.P., Daw, N.D., Dayan, P., O’Doherty, J.P.: States versus rewards: Dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron* 66, 585–595 (2010)
- Glimcher, P.W.: *Decisions, Uncertainty, and the Brain: The Science of Neuroeconomics*. MIT Press, Cambridge (2003)
- Glimcher, P.W., Rustichini, A.: Neuroeconomics: The consilience of brain and decision. *Science* 306, 447–452 (2004)
- Gluck, M.A.: Behavioral and neural correlates of error correction in classical conditioning and human category learning. In: Gluck, M.A., Anderson, J.R., Kosslyn, S.M. (eds.) *Memory and Mind: A Festschrift for Gordon H. Bower*, ch. 18, pp. 281–305. Lawrence Erlbaum Associates, New York (2008)
- Gold, J.I., Shadlen, M.N.: The neural basis of decision making. *Annual Review of Neuroscience* 30, 535–574 (2007)
- Goldman-Rakic, P.S.: Cellular basis of working memory. *Neuron* 14, 447–485 (1995)
- Goodnow, J.T.: Determinants of choice-distribution in two-choice situations. *The American Journal of Psychology* 68, 106–116 (1955)
- Gormezano, I., Schneiderman, N., Deaux, E.G., Fuentes, I.: Nictitating membrane: Classical conditioning and extinction in the albino rabbit. *Science* 138, 33–34 (1962)
- Grafton, S.T., Hamilton, A.F.: Evidence for a distributed hierarchy of action representation in the brain. *Human Movement Science* 26, 590–616 (2007)
- Graybiel, A.M.: The basal ganglia: learning new tricks and loving it. *Current Opinion in Neurobiology* 15, 638–644 (2005)
- Graybiel, A.M.: Habits, rituals, and the evaluative brain. *Annual Review of Neuroscience* 31, 359–387 (2008)
- Graybiel, A.M., Aosaki, T., Flaherty, A.W., Kimura, M.: The basal ganglia and adaptive motor control. *Science* 265, 1826–1831 (1994)
- Green, L., Myerson, J.: A discounting framework for choice with delayed and probabilistic rewards. *Psychological Bulletin* 130, 769–792 (2004)
- Gruen, R., Huber, M.: A framework for the development of robot behavior. In: 2005 AAAI Spring Symposium Series: Developmental Robotics. American Association for the Advancement of Artificial Intelligence, Palo Alto (2005)
- Gurney, K.: Reverse engineering the vertebrate brain: Methodological principles for a biologically grounded programme of cognitive modelling. *Cognitive Computation* 1, 29–41 (2009)
- Gurney, K., Prescott, T.J., Redgrave, P.: A computational model of action selection in the basal ganglia. I. A new functional anatomy. *Biological Cybernetics* 84, 401–410 (2001)
- Gurney, K., Prescott, T.J., Wickens, J.R., Redgrave, P.: Computational models of the basal ganglia: From robots to membranes. *Trends in Neuroscience* 27, 453–459 (2004)
- Haber, S.N.: The primate basal ganglia: Parallel and integrative networks. *Journal of Chemical Neuroanatomy* 26, 317–330 (2003)
- Haber, S.N., Kim, K.S., Mailly, P., Calzavara, R.: Reward-related cortical inputs define a large striatal region in primates that interface with associative cortical inputs, providing a substrate for incentive-based learning. *The Journal of Neuroscience* 26, 8368–8376 (2006)
- Haruno, M., Kawato, M.: Heterarchical reinforcement-learning model for integration of multiple cortico-striatal loops: fMRI examination in stimulus-action-reward association learning. *Neural Networks* 19, 1242–1254 (2006)
- Hazy, T.E., Frank, M.J., O’Reilly, R.C.: Neural mechanisms of acquired phasic dopamine responses in learning. *Neuroscience and Biobehavioral Reviews* 34, 701–720 (2010)
- Herrnstein, R.J.: Relative and absolute strength of response as a function of frequency of reinforcement. *Journal of the Experimental Analysis of Behavior* 4, 267–272 (1961)
- Hikosaka, O.: Basal ganglia mechanisms of reward-oriented eye movement. *Annals of the New York Academy of Science* 1104, 229–249 (2007)
- Hollerman, J.R., Schultz, W.: Dopamine neurons report an error in the temporal prediction of reward during learning. *Nature Neuroscience* 1, 304–309 (1998)
- Horvitz, J.C.: Mesolimbocortical and nigrostriatal dopamine responses to salient non-reward

- events. *Neuroscience* 96, 651–656 (2000)
- Houk, J.C., Wise, S.P.: Distributed modular architectures linking basal ganglia, cerebellum, and cerebral cortex: Their role in planning and controlling action. *Cerebral Cortex* 5, 95–110 (1995)
- Houk, J.C., Adams, J.L., Barto, A.G.: A model of how the basal ganglia generate and use neural signals that predict reinforcement. In: Houk, J.C., Davis, J.L., Beiser, D.G. (eds.) *Models of Information Processing in the Basal Ganglia*, ch. 13, pp. 249–270. MIT Press, Cambridge (1995)
- Houk, J.C., Bastianen, C., Fansler, D., Fishbach, A., Fraser, D., Reber, P.J., Roy, S.A., Simo, L.S.: Action selection and refinement in subcortical loops through basal ganglia and cerebellum. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 362, 1573–1583 (2007)
- Hull, C.L.: *Principles of Behavior*. Appleton-Century-Crofts, New York (1943)
- Humphries, M.D., Prescott, T.J.: The ventral basal ganglia, a selection mechanism at the crossroads of space, strategy, and reward. *Progress in Neurobiology* 90, 385–417 (2010)
- Ito, M., Doya, K.: Validation of decision-making models and analysis of decision variables in the rat basal ganglia. *The Journal of Neuroscience* 29, 9861–9874 (2009)
- Joel, D., Weiner, I.: The organization of the basal ganglia-thalamocortical circuits: Open interconnected rather than closed segregated. *Neuroscience* 63, 363–379 (1994)
- Joel, D., Niv, Y., Ruppert, E.: Actor-critic models of the basal ganglia: New anatomical and computational perspectives. *Neural Networks* 15, 535–547 (2002)
- Joshua, M., Adler, A., Bergman, H.: The dynamics of dopamine in control of motor behavior. *Current Opinion in Neurobiology* 19, 615–620 (2009)
- Kamin, L.J.: Predictability, surprise, attention, and conditioning. In: Campbell, B.A., Church, R.M. (eds.) *Punishment and Aversive Behavior*, pp. 279–296. Appleton-Century-Crofts, New York (1969)
- Kehoe, E.J., Schreurs, B.G., Graham, P.: Temporal primacy overrides prior training in serial compound conditioning of the rabbit's nictitating membrane response. *Animal Learning and Behavior* 15, 455–464 (1987)
- Kim, H., Sul, J.H., Huh, N., Lee, D., Jung, M.W.: Role of striatum in updating values of chosen actions. *The Journal of Neuroscience* 29, 14,701–14,712 (2009)
- Kishida, K.T., King-Casas, B., Montague, P.R.: Neuroeconomic approaches to mental disorders. *Neuron* 67, 543–554 (2010)
- Klopf, A.H.: *The Hedonistic Neuron: A Theory of Memory, Learning and Intelligence*. Hemisphere Publishing Corporation, Washington DC (1982)
- Kobayashi, S., Schultz, W.: Influence of reward delays on responses of dopamine neurons. *The Journal of Neuroscience* 28, 7837–7846 (2008)
- Konidaris, G.D., Barto, A.G.: Skill discovery in continuous reinforcement learning domains using skill chaining. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) *Advances in Neural Information Processing Systems (NIPS)*, vol. 22, pp. 1015–1023. MIT Press, Cambridge (2009)
- Lau, B., Glimcher, P.W.: Value representations in the primate striatum during matching behavior. *Neuron* 58, 451–463 (2008)
- Ljungberg, T., Apicella, P., Schultz, W.: Responses of monkey dopamine neurons during learning of behavioral reactions. *Journal of Neurophysiology* 67, 145–163 (1992)
- Ludvig, E.A., Sutton, R.S., Kehoe, E.J.: Stimulus representation and the timing of reward-prediction errors in models of the dopamine system. *Neural Computation* 20, 3034–3054 (2008)
- Maia, T.V.: Reinforcement learning, conditioning, and the brain: Successes and challenges. *Cognitive, Affective, and Behavioral Neuroscience* 9, 343–364 (2009)
- Maia, T.V., Frank, M.J.: From reinforcement learning models to psychiatric and neurobiological disorders. *Nature Neuroscience* 14, 154–162 (2011)
- Matsumoto, K., Suzuki, W., Tanaka, K.: Neuronal correlates of goal-based motor selection in the prefrontal cortex. *Science* 301, 229–232 (2003)
- Matsuzaka, Y., Picard, N., Strick, P.: Skill representation in the primary motor cortex after long-term practice. *Journal of Neurophysiology* 97, 1819–1832 (2007)

- McHaffie, J.G., Stanford, T.R., Stein, B.E., Coizet, V., Redgrave, P.: Subcortical loops through the basal ganglia. *Trends in Neurosciences* 28, 401–407 (2005)
- Middleton, F.A., Strick, P.L.: Basal-ganglia“projections” to the prefrontal cortex of the primate. *Cerebral Cortex* 12, 926–935 (2002)
- Miller, E.K., Cohen, J.D.: An integrative theory of prefrontal cortex function. *Annual Review of Neuroscience* 24, 167–202 (2001)
- Miller, J.D., Sanghera, M.K., German, D.C.: Mesencephalic dopaminergic unit activity in the behaviorally conditioned rat. *Life Sciences* 29, 1255–1263 (1981)
- Mink, J.W.: The basal ganglia: Focused selection and inhibition of competing motor programs. *Progress in Neurobiology* 50, 381–425 (1996)
- Mirolli, M., Mannella, F., Baldassarre, G.: The roles of the amygdala in the affective regulation of body, brain, and behaviour. *Connection Science* 22, 215–245 (2010)
- Montague, P.R., Dayan, P., Sejnowski, T.J.: A framework for mesencephalic dopamine systems based on predictive Hebbian learning. *Journal of Neuroscience* 16, 1936–1947 (1996)
- Montague, P.R., Hyman, S.E., Cohen, J.D.: Computational roles for dopamine in behavioural control. *Nature* 431, 760–767 (2004)
- Montague, P.R., King-Casas, B., Cohen, J.D.: Imaging valuation models in human choice. *Annual Review of Neuroscience* 29, 417–448 (2006)
- Moore, J.W., Choi, J.S.: Conditioned response timing and integration in the cerebellum. *Learning and Memory* 4, 116–129 (1997)
- Morris, G., Nevet, A., Arkadir, D., Vaadia, E., Bergman, H.: Midbrain dopamine neurons encode decisions for future action. *Nature Neuroscience* 9, 1057–1063 (2006)
- Mushiake, H., Saito, N., Sakamoto, K., Itoyama, Y., Tanji, J.: Activity in the lateral prefrontal cortex reflects multiple steps of future events in action plans. *Neuron* 50, 631–641 (2006)
- Nakahara, H., Itoh, H., Kawagoe, R., Takikawa, Y., Hikosaka, O.: Dopamine neurons can represent context-dependent prediction error. *Neuron* 41, 269–280 (2004)
- Ng, A., Harada, D., Russell, S.: Policy invariance under reward transformations: theory and applications to reward shaping. In: *Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 278–287 (1999)
- Nicola, S.M.: The nucleus accumbens as part of a basal ganglia action selection circuit. *Psychopharmacology* 191, 521–550 (2007)
- Niv, Y.: Reinforcement learning in the brain. *Journal of Mathematical Psychology* 53, 139–154 (2009)
- Niv, Y., Duff, M.O., Dayan, P.: Dopamine, uncertainty, and TD learning. *Behavioral and Brain Functions* 1, 6 (2005)
- Niv, Y., Daw, N.D., Dayan, P.: Choice values. *Nature Neuroscience* 9, 987–988 (2006a)
- Niv, Y., Joel, D., Dayan, P.: A normative perspective on motivation. *Trends in Cognitive Sciences* 10, 375–381 (2006b)
- Nomoto, K., Schultz, W., Watanabe, T., Sakagami, M.: Temporally extended dopamine responses to perceptually demanding reward-predictive stimuli. *The Journal of Neuroscience* 30, 10,692–10,702 (2010)
- O’Doherty, J.P., Dayan, P., Schultz, J., Deichmann, R., Friston, K., Dolan, R.J.: Dissociable roles of ventral and dorsal striatum in instrumental conditioning. *Science* 304, 452–454 (2004)
- Olds, J., Milner, P.: Positive reinforcement produced by electrical stimulation of septal area and other regions of rat brain. *Journal of Comparative and Physiological Psychology* 47, 419–427 (1954)
- O’Reilly, R.C., Frank, M.J.: Making working memory work: a computational model of learning in the prefrontal cortex and basal ganglia. *Neural Computation* 18, 283–328 (2006)
- Packard, M.G., Knowlton, B.J.: Learning and memory functions of the basal ganglia. *Annual Review of Neuroscience* 25, 563–593 (2002)
- Pasupathy, A., Miller, E.K.: Different time courses of learning-related activity in the prefrontal cortex and striatum. *Nature* 433, 873–876 (2005)
- Pavlov, I.P.: *Conditioned Reflexes: An Investigation of the Physiological Activity of the Cerebral Cortex*. Oxford University Press, Toronto (1927)

- Pennartz, C.M., Berke, J.D., Graybiel, A.M., Ito, R., Lansink, C.S., van der Meer, M., Redish, A.D., Smith, K.S., Voorn, P.: Corticostriatal interactions during learning, memory processing, and decision making. *The Journal of Neuroscience* 29, 12,831–12,838 (2009)
- Pessiglione, M., Seymour, B., Flandin, G., Dolan, R.J., Frith, C.D.: Dopamine-dependent prediction errors underpin reward-seeking behaviour in humans. *Nature* 442, 1042–1045 (2006)
- Phelps, E.A., LeDoux, J.E.: Contributions of the amygdala to emotion processing: From animal models to human behavior. *Neuron* 48, 175–187 (2005)
- Poldrack, R.A., Sabb, F.W., Foerde, K., Tom, S.M., Asarnow, R.F., Bookheimer, S.Y., Knowlton, B.J.: The neural correlates of motor skill automaticity. *The Journal of Neuroscience* 25, 5356–5364 (2005)
- Pompilio, L., Kacelnik, A.: State-dependent learning and suboptimal choice: when starlings prefer long over short delays to food. *Animal Behaviour* 70, 571–578 (2005)
- Redgrave, P., Gurney, K.: The short-latency dopamine signal: a role in discovering novel actions? *Nature Reviews Neuroscience* 7, 967–975 (2006)
- Redgrave, P., Gurney, K., Reynolds, J.: What is reinforced by phasic dopamine signals? *Brain Research Reviews* 58, 322–339 (2008)
- Redgrave, P., Rodriguez, M., Smith, Y., Rodriguez-Oroz, M.C., Lehericy, S., Bergman, H., Agid, Y., DeLong, M.R., Obeso, J.A.: Goal-directed and habitual control in the basal ganglia: implications for Parkinson's disease. *Nature Reviews Neuroscience* 11, 760–772 (2010)
- Redish, A.D., Jensen, S., Johnson, A.: A unified framework for addiction: Vulnerabilities in the decision process. *Behavioral and Brain Sciences* 31, 415–487 (2008)
- Rescorla, R.A., Wagner, A.R.: A theory of pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In: Black, A.H., Prokasy, W.F. (eds.) *Classical Conditioning II: Current Research and Theory*, pp. 64–99. Appleton-Century-Crofts, New York (1972)
- Richardson, W.K., Warzak, W.J.: Stimulus stringing by pigeons. *Journal of the Experimental Analysis of Behavior* 36, 267–276 (1981)
- Roesch, M.R., Calu, D.J., Schoenbaum, G.: Dopamine neurons encode the better option in rats deciding between differently delayed or sized rewards. *Nature Neuroscience* 10, 1615–1624 (2007)
- Roesch, M.R., Singh, T., Brown, P.L., Mullins, S.E., Schoenbaum, G.: Ventral striatal neurons encode the value of the chosen action in rats deciding between differently delayed or sized rewards. *The Journal of Neuroscience* 29, 13,365–13,376 (2009)
- Samejima, K., Doya, K.: Multiple representations of belief states and action values in corticobasal ganglia loops. *Annals of the New York Academy of Sciences* 1104, 213–228 (2007)
- Samejima, K., Ueda, Y., Doya, K., Kimura, M.: Representation of action-specific reward values in the striatum. *Science* 310, 1337–1340 (2005)
- Satoh, T., Nakai, S., Sato, T., Kimura, M.: Correlated coding of motivation and outcome of decision by dopamine neurons. *The Journal of Neuroscience* 23, 9913–9923 (2003)
- Schultz, W.: Responses of midbrain dopamine neurons to behavioral trigger stimuli in the monkey. *Journal of Neurophysiology* 56, 1439–1461 (1986)
- Schultz, W.: Predictive reward signal of dopamine neurons. *Journal of Neurophysiology* 80, 1–27 (1998)
- Schultz, W.: Behavioral theories and the neurophysiology of reward. *Annual Review of Psychology* 57, 8–115 (2006)
- Schultz, W.: Multiple dopamine functions at different time courses. *Annual Review of Neuroscience* 30, 259–288 (2007)
- Schultz, W.: Dopamine signals for reward value and risk: basic and recent data. *Behavioral and Brain Functions* 6, 24 (2010)
- Schultz, W., Apicella, P., Ljungberg, T.: Responses of monkey dopamine neurons to reward and conditioned stimuli during successive steps of learning a delayed response task. *The Journal of Neuroscience* 13, 900–913 (1993)
- Schultz, W., Dayan, P., Montague, P.R.: A neural substrate of prediction and reward. *Science* 275, 1593–1599 (1997)

- Schultz, W., Tremblay, L., Hollerman, J.R.: Changes in behavior-related neuronal activity in the striatum during learning. *Trends in Neuroscience* 26, 321–328 (2003)
- Seger, C.A., Miller, E.K.: Category learning in the brain. *Annual Review of Neuroscience* 33, 203–219 (2010)
- Selfridge, O.J., Sutton, R.S., Barto, A.G.: Training and tracking in robotics. In: Joshi, A. (ed.) *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pp. 670–672. Morgan Kaufmann, San Mateo (1985)
- Shah, A.: Biologically-based functional mechanisms of motor skill acquisition. PhD thesis, University of Massachusetts Amherst (2008)
- Shah, A., Barto, A.G.: Effect on movement selection of an evolving sensory representation: A multiple controller model of skill acquisition. *Brain Research* 1299, 55–73 (2009)
- Shanks, D.R., Tunney, R.J., McCarthy, J.D.: A re-examination of probability matching and rational choice. *Journal of Behavioral Decision Making* 15, 233–250 (2002)
- Siegel, S., Goldstein, D.A.: Decision making behaviour in a two-choice uncertain outcome situation. *Journal of Experimental Psychology* 57, 37–42 (1959)
- Skinner, B.F.: *The Behavior of Organisms*. Appleton-Century-Crofts, New York (1938)
- Staddon, J.E.R., Cerutti, D.T.: Operant behavior. *Annual Review of Psychology* 54, 115–144 (2003)
- Sutton, R.S.: Learning to predict by methods of temporal differences. *Machine Learning* 3, 9–44 (1988)
- Sutton, R.S., Barto, A.G.: Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review* 88, 135–170 (1981)
- Sutton, R.S., Barto, A.G.: A temporal-difference model of classical conditioning. In: *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, pp. 355–378 (1987)
- Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press, Cambridge (1998)
- Tanji, J., Hoshi, E.: Role of the lateral prefrontal cortex in executive behavioral control. *Physiological Reviews* 88, 37–57 (2008)
- Thorndike, E.L.: *Animal Intelligence: Experimental Studies*. Macmillan, New York (1911)
- Tindell, A.J., Berridge, K.C., Zhang, J., Pecina, S., Aldridge, J.W.: Ventral pallidal neurons code incentive motivation: Amplification by mesolimbic sensitization and amphetamine. *European Journal of Neuroscience* 22, 2617–2634 (2005)
- Tobler, P.N., Dickinson, A., Schultz, W.: Coding of predicted reward omission by dopamine neurons in a conditioned inhibition paradigm. *The Journal of Neuroscience* 23, 10,402–10,410 (2003)
- Tobler, P.N., Fiorillo, C.D., Schultz, W.: Adaptive coding of reward value by dopamine neurons. *Science* 307, 1642–1645 (2005)
- Tolman, E.C.: Cognitive maps in rats and men. *The Psychological Review* 55, 189–208 (1948)
- Tolman, E.C.: There is more than one kind of learning. *Psychological Review* 56, 44–55 (1949)
- Waelti, P., Dickinson, A., Schultz, W.: Dopamine responses comply with basic assumptions of formal learning theory. *Nature* 412, 43–48 (2001)
- Wallis, J.D.: Orbitofrontal cortex and its contribution to decision-making. *Annual Review of Neuroscience* 30, 31–56 (2007)
- Watson, J.B.: *Behavior: An Introduction to Comparative Psychology*. Holt, New York (1914)
- Wickens, J.R.: Synaptic plasticity in the basal ganglia. *Behavioural Brain Research* 199, 119–128 (2009)
- Wickens, J.R., Budd, C.S., Hyland, B.I., Arbuthnott, G.W.: Striatal contributions to reward and decision making. Making sense of regional variations in a reiterated processing matrix. *Annals of the New York Academy of Sciences* 1104, 192–212 (2007)
- Widrow, B., Hoff, M.E.: Adaptive switching circuits. In: 1960 WESCON Convention Record Part IV, pp. 96–104. Institute of Radio Engineers, New York (1960)
- Wilson, C.J.: Basal ganglia. In: Shepherd, G.M. (ed.) *The Synaptic Organization of the Brain*, ch. 9, 5th edn., pp. 361–414. Oxford University Press, Oxford (2004)

- Wise, R.A.: Dopamine, learning and motivation. *Nature Reviews Neuroscience* 5, 483–494 (2004)
- Wolpert, D.: Probabilistic models in human sensorimotor control. *Human Movement Science* 27, 511–524 (2007)
- Wörgötter, F., Porr, B.: Temporal sequence learning, prediction, and control: A review of different models and their relation to biological mechanisms. *Neural Computation* 17, 245–319 (2005)
- Wrase, J., Kahnt, T., Schlagenhauf, F., Beck, A., Cohen, M.X., Knutson, B., Heinz, A.: Different neural systems adjust motor behavior in response to reward and punishment. *NeuroImage* 36, 1253–1262 (2007)
- Wyvell, C.L., Berridge, K.C.: Intra-accumbens amphetamine increases the conditioned incentive salience of sucrose reward: Enhancement of reward “wanting” without enhanced “liking” or response reinforcement. *Journal of Neuroscience* 20, 8122–8130 (2000)
- Yin, H.H., Ostlund, S.B., Balleine, B.W.: Reward-guided learning beyond dopamine in the nucleus accumbens: the integrative functions of cortico-basal ganglia networks. *European Journal of Neuroscience* 28, 1437–1448 (2008)
- Yu, A., Dayan, P.: Uncertainty, neuromodulation and attention. *Neuron* 46, 681–692 (2005)

游戏领域的强化学习

István Szita

摘要

强化学习和游戏有着一段长期互利的共同历史。从一方面来看，游戏是测试强化学习算法的一个丰富而具有挑战性的领域。另一方面，在几个游戏中，最优秀的电脑玩家使用的都是强化学习。本章从精选的游戏及著名的强化学习实现开始。不做任何修改的基本的强化学习算法对于高级玩法的游戏通常是不够的，因此，对于额外的想法（领域知识植入的方式，及为具备良好扩展性而在实现上的考虑）的讨论是很有必要的。仔细审查这些细节以了解其潜力及局限性。17.2 节列出了强化学习在游戏中遇到的挑战以及对已有解决方案的评论。虽然这个列表的视角是以游戏为中心的，而且有些项目是特定于游戏的（比如对手建模），但概述中的绝大部分也可以为其他类型的应用提供一些见解。17.3 节将简述强化学习如何在游戏开发中发挥作用并应用到商业电子游戏中。最后，17.4 节提供了更多对具体游戏及其解决方案的深入讨论和建议。

17.1 简介

强化学习（RL）和游戏具有悠久而富有成效的共同历史。最早的学习程序之一——Samuel 的跳棋玩家（Checkers）已经具有时序差分（TD）学习的思想，比真正提出和分析 TD 学习早了几十年。而在另一种游戏西洋双陆棋（Backgammon）中，当 Tesauro 的 TD-Gammon 达到并超过顶级人类玩家的水平并且是完全通过自己学习时，强化学习第一次取得了极大的成功。从那时起，强化学习已应用于许多其他的游戏中，虽然不能在每个游戏中都重复 TD-Gammon 的成功，但是仍出现了不错的效果以及许多经验教训。我们希望在本章中介绍它们在传统游戏和电脑游戏中的应用，不仅有即时战略游戏、第一人称射击游戏，还有角色扮演游戏。最值得注意的是，强化学习方法似乎霸占了人工智能研究的旗舰应用之一——围棋（Go）。

539

从另一个角度来看，游戏是强化学习研究的优秀测试平台。游戏旨在娱乐和挑战人类，因此，通过学习游戏，我们可以（有希望）了解人工智能以及人工智能需要解决的挑战。同时，游戏也是对强化学习算法来说具有挑战性的领域，可能是由于它们对人类来说也一样具有挑战性：游戏旨在挑战玩家做出有趣的决定。挑战的类型有很多变化，我们的目标是对这些精选出的挑战及其强化学习解决方案予以介绍。

17.1.1 目标和结构

本章的一个目标是收集用于游戏的著名强化学习应用程序。但是另一个更重要的目标是了解强化学习算法如何（以及为什么）在实践中有效（或失效）。本章中提到的多数算法在本

书的其他章节中有详细介绍。这些算法的理论分析(如果存在)对它们能在理想条件下工作提供了保证,但对于大多数游戏来说这些分析是不切实际的,因为它们的条件过于限制、声明过于松散或者两者皆有。例如,我们知道,如果环境是有限马尔可夫决策过程(MDP),每个状态的值单独存储、学习率恰当地递减,而且探索得比较充分,那么TD学习[⊖]能收敛到最优策略。上述多数条件在典型游戏应用程序中会被违反。然而,TD学习在西洋双陆棋中非常成功,而在其他游戏(例如,俄罗斯方块)中并不理想。有很多文献试图识别出使TD学习及其他强化学习算法表现良好的游戏属性。我们认为,对这些尝试进行概述会非常有助于未来的发展。

在强化学习的任何应用中,算法的选择只是决定成功或失败的因素之一。通常算法的选择甚至不是最重要的因素,因为表示的选择、形式化、领域知识的编码、额外的启发信息和变体以及恰当的参数设置,都会有很大的影响。对于以上每个问题,我们可以找到为了征服特定游戏而开发的令人兴奋的想法。不幸的是(但也不足为奇),没有“灵丹妙药”(magic bullet):所有的方法都或多或少是针对特定游戏或类型的。不过我们认为,研究这些想法可以为我们提供有用的见解,并且其中一些发现可以推广到其他应用。

游戏的多样化使得话题也多种多样,不易组织。为了保持一致性,我们先从一些深入研究过的游戏中了解强化学习:西洋双陆棋、国际象棋、围棋、扑克(Poker)、俄罗斯方块(Tetris)和即时战略游戏。这些游戏可以作为案例研究,它们引入了许多不同的强化学习问题。17.3节总结了一些重要的问题和挑战。17.4节可能是最值得关注的部分,因为它综述了强化学习在游戏中的实际应用。最后,17.5节给出了进一步阅读的参考文献以及我们的结论。

17.1.2 范围

在关于游戏中强化学习应用的章节中,必须沿着两个维度划定讨论的边界:包含什么样的算法?以及包含什么样的游戏?这两个决定都不容易。

显然,TD-Gammon就是在这个范围内(而深蓝可能不是)。但是,应用于围棋的UCT是否属于这里?虽然UCT与强化学习显然有很强的联系,但我们可以将其放在规划甚至游戏理论的范畴内。其他应用也是这种模糊的状态(另一个很好的例子是使用演化方法处理强化学习任务)。按照本书的理论,我们考虑广义上的“强化学习应用”,即包括从强化学习的要素中获得灵感的所有方法。然而,我们不得不放弃一些重要的游戏,例如扑克、桥牌(Bridge)或者PSPACE-hard的推箱子(Soko-ban),以及一般和对弈(general-sum game)的理论(它激发了AI研究中的大量成果,但其中大部分一般不认为是强化学习)。

对于游戏,我们不希望排除人们玩的任何实际的游戏,所以电脑游戏、现代棋盘游戏肯定会包含在内。然而,本章将不可避免地偏向“传统”游戏,这仅仅是因为有更多的研究成果。

17.2 游戏展示厅

在本节中,我们将概述几个游戏在强化学习相关方面的结果。我们专注于从强化学习的角度看待它们。所以,我们对游戏规则的解释仅仅到理解特定的强化学习问题所必需的细节

⊖ 在理论工作中,TD学习是指一种策略评估方法,而在游戏相关的文献中,它是指“一个让评论家使用TD学习的演员-评论家(actor-critic)学习”。

的程度。

541

17.2.1 西洋双陆棋

在西洋双陆棋比赛中, 两名玩家尽可能快地从棋盘上移除他们的棋子。棋盘上有 24 个线性排列的区域, 每个玩家起始时有 15 个对称排列的棋子。图 17.1 展示了棋盘的起始状态。玩家需要往相反的方向移动他们的棋子。玩家的某个棋子到达路径的尽头后, 只有当该玩家的所有棋子都已经在最后一个大区时, 它才可以被移除。两名玩家轮流进行游戏, 每轮掷两个骰子。移动步数由骰子确定, 但玩家可以决定移动哪个棋子。单个的棋子可以被击中 (hit)(之后需要从起点开始), 但是玩家不能移动到具有两个或更多对手棋子的位置^①。

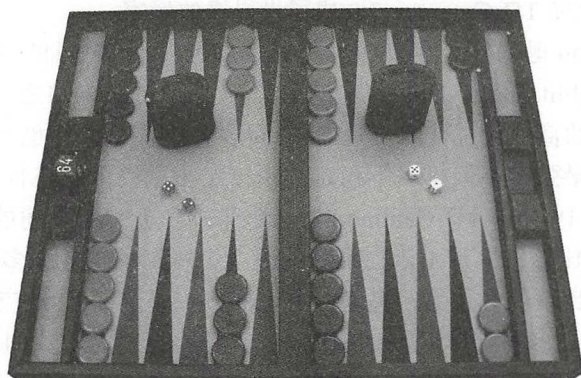


图 17.1 西洋双陆棋棋盘初始状态, 包含两套骰子和一个倍数方块

基本的策略包括封锁关键位置、建立难以跳过的长块障碍, 还有快速到达终点。在更高的层次上, 玩家需要相当准确地估计各种事件 (相对于其他事件) 的概率。另一个策略要素是“倍数方块” (doubling cube): 任何玩家都可以选择加倍游戏赌注。另一名玩家有两种选择: 可以接受加倍并获得方块, 有权提出下一次加倍; 或者可以放弃游戏, 如果是这样只会失去目前的赌注。知道何时提出加倍以及何时接受是西洋双陆棋最难的部分。

由于西洋双陆棋的偶然性, 传统的搜索算法对它是无效的: 对于每一个回合, 掷出的骰子有 21 个可能的结果, 每个平均有 20 种合法的移动。这就让分支因子超过了 400, 使深度前瞻搜索 (lookahead search) 变得不可能。

542

17.2.1.1 TD-Gammon

TD-Gammon 的第一个版本 [Tesauro, 1992, 2002] 使用神经网络作为位置评估器。网络的输入是游戏位置的编码, 输出是位置的价值^②。抛掷骰子后, 程序会尝试所有合法的移动并评估移动后的位置, 然后采用评分最高的移动。网络没有考虑加倍的可能性, 而这是由一个单独的启发式公式处理的。胜利时奖励 +1, 失败时为 0^③, 输出获胜的近似概率值。对于每个玩家, 每个棋盘位置由使用截断一元编码 (如果该区域具有至少 k 个棋子, 则输入 k

① 关于规则的详细描述, 请参考互联网上的西洋双陆棋资源, 例如, <http://www.play65.com/Backgammon.html>。

② 实际上有四个输出神经元, 每一个对应于双方玩家在有 / 没有 “gammon” (当一个玩家在另一个玩家开始移除他自己的棋子之前赢了) 这种情况的一个组合。为了简单起见, 我们只考虑两个玩家都没有 gammon 的情况。

③ Tesauro 考虑的情况稍微复杂一些: 西洋双陆棋的某些胜利价值为 +2, 甚至某些罕见的情况为 +3; 此外, 点值可以多次加倍。

的编码为 1, 否则为 0) 的四个神经元表示。TD-Gammon 后来的版本还使用了专家特征编码作输入, 例如“点数”(pip count), 这是一种启发式的进度度量(棋子离目标的总距离)。

执行每个步骤后, 根据 TD 规则更新神经元的输出, 并通过反向传播(Backpropagation)规则更新权值。该网络是通过自我对弈(self-play)和非显式探索训练的。

TD-Gammon 的效果非常好。Tesauro 之前的西洋双陆棋程序 Neurogammon 使用了具有相同架构的神经网络, 但使用的是由人类西洋双陆棋专家标记的样本进行训练。TD-Gammon 明显比 Neurogammon 健壮得多, 即使在残缺的情况下(只接受原始的表格表示作输入而没有领域知识编码特征), 它也能媲美 Neurogammon。在后来的版本中, Tesauro 增加了隐藏层神经元的数量、训练游戏的轮数(达到 1 500 000), 还增加了 3 层的前瞻, 并且改进了表示。这些变化使 TD-Gammon 3.0 成了世界级的玩家。

此后, TD-Gammon 退隐了, 但 TD(λ) 仍然是如今几个最强的西洋双陆棋游戏程序 Snowie、Jellyfish、Bgblitz 和 GNUbg 的基础。关于它们的细节知之甚少, 因为除 GNUbg 之外的程序都是闭源的商业产品(即使是 GNUbg, 它的技术文档也不太好)。有一个共识是, Snowie 是它们中最强的, 但排名不稳定。由于结果的高度随机性, 不好与人类做比较(一个典型的例子: 在 1998 年 TD-Gammon 与马尔科姆·戴维斯之间的 100 场比赛中, TD-Gammon 赢得了 99 场比赛, 但是由于在唯一的败场中输掉的点数太多, 它仍然输掉了整个比赛)。作为许多游戏比赛的替代品, 快速走子(rollout)分析广泛地应用于比较玩家。一个西洋双陆棋电脑程序试图通过从当前状态模拟数千场比赛来为游戏的每一步找到最佳的移动, 并计算玩家的实际选择与它的差距。依照快速走子分析的西洋双陆棋程序已经超过了最好的人类玩家的实力 [Tesauro, 2002]。

543

17.2.1.2 为什么 TD-Gammon 有效?

TD-Gammon 的成功对于 TD(λ) 的流行绝对起到了极大促进作用。TD 学习已经开发了包括游戏在内的许多应用(概述参见 [Ghory, 2004])。在许多这种游戏中, TD 并不能重复同样的成功, 这更显得 TD-Gammon 的表现惊人。西洋双陆棋的 TD 学习必须以完全的通用性来处理强化学习问题: 存在随机转换、随机且延迟的奖励以及巨大的状态空间。由于是自我对弈, 其环境并不是真正的 MDP (对手在不断变化)。更令人不安的是使用非线性函数逼近。此外, TD-Gammon 没有进行任何探索, 它总是做贪婪选择, 这可能导致性能不佳。包括 [Tesauro, 1995, 1998; Pollack and Blair, 1997; Ghory, 2004; Wiering, 2010], 许多作者都试图找出 TD 在西洋双陆棋中成功的原因。我们尝试在这里总结他们的论据。

表示。正如本书开头的章节所强调的那样, 表示的恰当选择对于任何强化学习应用都至关重要。Tesauro 声称, TD-Gammon 的状态表示是“无知识”的, 因为它可以不使用任何游戏知识来产生。另一方面, 截断的一元表示似乎很适合西洋双陆棋, 就如同天然的棋盘表示一样(每个区域和玩家用一个单位表示)。当然, 增加更多有益的特征能显著增强玩家实力。事实上, 使用相同特征但没有强化学习的 Neurogammon 也是一个相当不错的电脑玩家。

随机性。众所周知, 游戏的随机性使学习任务变得更容易。首先, 它能让价值函数平滑: “相似”的状态值彼此接近, 因为随机波动可以掩盖彼此的差异。函数近似对于平滑目标函数更有效。其次, 随机性有助于解决探索的困境, 或者如本例一样完全避免它(在 TD-Gammon 中, 学习器一直做贪婪的选择)。由于骰子的作用, 玩家将无需在探索上做额外花费就可以访问状态空间的绝大部分。第三个因素是, 据称人类玩家不擅长概率估计, 这使其更容易失败。

训练机制。TD-Gammon 通过自我对弈来训练。一般来说,这可能既有好处也有坏处:学习器能与同一级别对手对弈,而另一方面,学习过程可能会卡住,然后学习器可能会产生一个与弱者(它本身)对弈的特殊策略,导致其永远不会遇到一个强大的对手。17.3.3.1 节将详细分析自我对弈的优点和缺点,但在这里我们需要注意,西洋双陆棋的“自动探索”属性有助于防止学习过程卡住:即使对手很弱,它也可能幸运地到达一个强大的状态。这为学习的平衡性创造了动力。 [544]

随机性和“自动探索”使得西洋双陆棋在各类游戏中相当独特。实际上 [Pollack and Blair, 1997] 认为,西洋双陆棋的这些特殊性质比 TD 学习和神经网络对成功的贡献更大。他们使用简单的梯度下降来训练一个线性结构,达到了相当好的性能。另一方面, [Tesauro, 1998] 声称这还不够,因为神经网络实际上发现了线性结构无法捕获的重要特征。

参数。TD(λ) 的参数 λ 决定了自举的总量。普遍的共识是, λ 的中间值比极端值 0 (完全自举) 或 1 (无自举) 都更好。Tesauro 在初始的实验中使用 $\lambda = 0.7$, 但之后切换到 $\lambda = 0$, 因为他注意到性能没有显著的差异, 并且没有充分的迹象表明 TD 更新规则的计算更简单快速。另一方面, [Wiering, 2010] 报告说, 初始的学习使用较高的 λ (≈ 0.8) 学习进度最快, 而较低的值 λ (但仍然 ≈ 0.6) 从长远来看是最佳的。

恰当的架构。[Tesauro, 2002] 描述了该架构的其他部分。我们在此强调一点: 时序差分学习是 TD-Gammon 成功的重要因素, 但知道如何使用它同样重要。通过训练神经网络来预测给定状态的胜利概率, 其预测不是非常准确, 偏差高达 0.1, 对于直接使用来说太高了。幸运的是, 在相似的游戏状态中, 偏差的变化不大, 所以神经网络预测仍然可以用来比较和排列给定位置的合法移动的价值, 因为大多数时候相对值也是可靠的。然而, 我们需要知道处理倍数方块的确切胜利概率, 即决定何时提出加倍赌注以及何时接受加倍。因此, TD-Gammon 使用独立的启发式公式来做加倍的决策。在某些情况下可以精确地计算某些事件的概率 (例如, 终结游戏的情况或者通过某个障碍)。TD-Gammon 后来的版本使用预先计算的概率作为输入特征, 而如今领先的西洋双陆棋程序可能同时使用了预先计算的概率和终结游戏的数据库。

17.2.2 国际象棋

国际象棋是一个典型的双人、全信息的零和 (zero-sum) 游戏^①。它的人气、声望和复杂性使它成为数十年来人工智能研究的首要目标。在其著名的 1997 年锦标赛中, 深蓝击败了当时的世界冠军加里·卡斯帕罗夫 [Hsu, 2002]。从那时起, 国际象棋程序已经远远超过了最厉害的人类: 现在 FIDE 排名的领先者 (Magnus Carlsen) 评分为 2826 Elo, 而计算机象棋奥林匹克的冠军 Rybka 应该远高于 3000^②。

国际象棋是目前为止强化学习没有非常成功

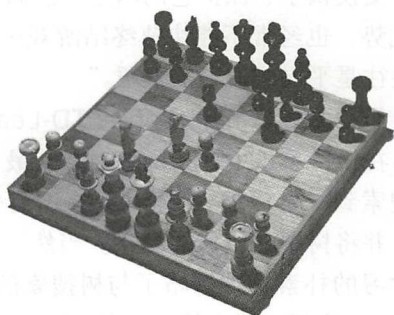


图 17.2 Sicilian 开局的棋盘 (来源: Wikimedia Commons, 作者: Rmrfstar) [545]

① 详细规则参见 http://en.wikipedia.org/wiki/Rules_of_chess#Play_of_the_game。

② Rybka 在 CCRL 的电脑棋手排名中的评分超过 3200 (见 http://computerchess.org.uk/ccrl/4040/rating_list_all.html), 但包含不同玩家库的 Elo 排名之间不能直接进行比较。

地应用而且不缺乏尝试的一线游戏之一。强化学习在国际象棋中的初步尝试包括 [Gherry, 1993], 其应用 Q 学取得了非常有限的成功。我们在这里讨论两个程序, [Thrun, 1995] 的 NeuroChess 和 [Baxter et al, 2000] 的 TD-Leaf。

17.2.2.1 NeuroChess

[Thrun, 1995] 的 NeuroChess 训练了两个神经网络, 一个代表价值函数 V , 另一个表示预测模型 M 。对于任意游戏位置 s , $M(s)$ 是两个半步后游戏状态的预期编码的近似值。每个棋盘都被编码为一个通过手动编码特征的 175 个元素的向量 (可能包含象棋评估功能的标准特征, 例如, 棋子价值、棋子位置价值、叠兵的惩罚等), 评估函数网络通过时序差分法的一种有趣延伸进行训练: 不仅将价值函数调整为目标值, 而且还将其斜率调整为目标价值函数的斜率。这应该能实现更好的拟合。令 s_1, s_2, \dots 为白棋回合的状态序列 (可以类似地处理黑棋的回合), 并考虑时间步长 t 。如果 t 是最后一步, 则 $V(s_t)$ 的目标值是游戏的最终结果 (0 或 ± 1)。根据 TD(0) 的更新规则, 对于不是最后一步的 t , 目标值为 $V^{\text{target}} = \gamma V(x_{t+1})$ (注意所有即时奖励为 0)。目标斜率为

$$dV^{\text{target}} = \frac{\partial V^{\text{target}}}{\partial x_t} = \gamma \frac{\partial V(x_{t+1})}{\partial x_t} = \gamma \frac{\partial V(x_{t+1})}{\partial x_{t+1}} \frac{\partial x_{t+1}}{\partial x_t} \approx \gamma \frac{\partial V(x_{t+1})}{\partial x_{t+1}} \frac{\partial M(x_t)}{\partial x_t}$$

V 和 M 的斜率可以很容易地从神经网络中提取出来。预测模型通过 120 000 场大师级游戏的 (有监督) 反向传播进行训练, 而 V 则使用叫作 Tangent-Prop 的算法, 使用目标值及其斜率进行时序差分训练。TD 训练持续了 2400 场。该架构的其他细节未知, 但一些提高性能的技巧值得一提。根据 Thrun 的说法, 其中有两个技巧最具影响力。首先, 每个游戏都来自游戏数据库的随机的步骤数目开始, 然后通过自我对弈完成。初始化确保了学习的精力集中在有趣的的游戏情况之上。其次, 设计的特征与原始表示相比能更平滑地表示棋盘 (即相似游戏位置的特征向量通常彼此接近)。更进一步的技巧包括静态搜索[⊖]、折扣 ($\gamma = 0.98$) 以及最终状态的学习率的增加。

所得到的玩家明显优于通过纯 TD 学习训练得到的 (没有斜率信息或预测模型)。然而, 即使 GNUchess 的搜索深度限制在两个半步, NeuroChess 与 GNUchess 对弈也只赢得了 13% 的比赛, 它仍然不是一个强大的玩家。据 Thrun 说: “NeuroChess 已经成功地学习了保护棋子、交换棋子、保护它的国王。然而, 它还没有学会以协调的方式开局, 而且即使具有棋子的优势, 也经常不能快速终结游戏……最重要的是, 它的开局仍然是令人难以置信的糟糕, 这往往是平局或败局的根源。”

17.2.2.2 KnightCap 和 TD-Leaf(λ)

在国际象棋中评估函数通常与最大搜索或其他多步前瞻算法相结合。这样的算法将游戏树搜索到一定深度 d (不一定在所有分支上是均匀的), 用启发式评估函数 V 评估最深的节点, 并将树中的值传播到根。当然, 评估函数可以用 TD 或其他强化学习方法训练。时序差分学习的朴素实现忽略了与树搜索的交互: 在 t 时刻, 它尝试将当前状态 $V(x_t)$ 的值调整为 $V(x_{t+1})$ 。然而, 搜索算法不使用 $V(x_t)$, 但状态 d 的值在搜索树中下降。TDLeaf(λ) 的想法是使用搜索树提供的额外信息。

⊖ 静态搜索 (quiescence search) 在“感兴趣”或“非安静”移动的附近扩展搜索树以考虑其长期影响。例如, 在树的最深层上, 捕获对手的棋子是非常有利的, 即使是单层的前瞻也可以揭露其后的反捕获 (counter-capture)。

为了了解 TD-Leaf 如何工作, 我们考虑一个以 x_t 为根的游戏树, 包含所有深度最多为 d 的移动, 所有节点都用启发式函数 V 进行评估。树的主变化 (principal variation) 是指从根开始的路径, 每个玩家根据 V 选择极值最优的移动。令 x_t 表示主变化的叶节点。TD-Leaf 将时序差分学习应用于以下主叶节点: 使用 $TD(\lambda)$ 更新规则, 将 $V(x_t)$ 变为 $V(x_{t+1})$ 。其有用的原因是启发式评估函数恰好在这些主节点中使用 (尽管是与其他节点相比)。我们注意到, TD-Leaf 可以看作多步的 TD 学习, 其中多步的前瞻是通过改进的策略完成的。

KnightCap[Baxter et al, 2000] 使用 $TD\text{-}Leaf(\lambda)$ 来训练一个国际象棋的评估函数。评估函数是 5872 个手动编码的特征的线性组合。特征分为四组, 分别用于游戏的不同阶段 (开局、中场、结局和配对), 包括棋子的子力优势、棋子的位置优势以及其他标准特征。该算法使用 $\lambda = 0.7$ 的合格轨迹及非常高的学习率 $\alpha = 1.0$ (这意味着旧的值会被立即忘记并覆盖)。KnightCap 尝试分离对手的影响: 正的 TD 误差值可能意味着对手有失误[⊖], 故 KnightCap 只有在可以预测对手的动作时才会更新 (因此这似乎是一个合理的举动)。此外, 为了使学习向前执行, 子力的权值必须初始化为默认值。 [547]

KnightCap 成功的关键因素是训练机制。根据 [Baxter et al, 2000] 的研究结论, 自我对弈会过早收敛且最终的表现很差。因此, 他们让 KnightCap 在一个国际象棋互联网服务器上与各个水平的人类对弈。服务器通常匹配具有大致相等排名的玩家, 这导致了一个有趣的效果: KnightCap 了解到, 它不断地与更好的对手对弈, 这为进一步改进提供了足够的经验。最终, KnightCap 的 Elo 排名从初始的 1650 攀升到略低于人类大师级水平的 2150。使用额外的开局库, Elo 排名升到了 2400 ~ 2500, 达到了合理的水平。

我们注意到, TD-Leaf 已经成功应用于其他游戏, 其中最著名的是 [Schaeffer et al, 2001] 的 Checkers, 其学习的权值集与当时最好的人工玩家相当。

17.2.2.3 TreeStrap (minimax) 和 TreeStrap ($\alpha\beta$)

[Veness et al, 2009] 修改了 TD-Leaf 的思想: TD-Leaf 使用步骤 t 的主节点来更新步骤 $t-1$ 的主节点, 而 TreeStrap (minimax) 用于在时序 t 更新搜索树。此外, 不仅根节点要更新, 搜索树的每个内部节点都要更新为其相应的主节点。TreeStrap ($\alpha\beta$) 使用了同样的技巧, 但是由于剪枝, 下层节点的值并不总是确定的, 只能界定在一个区间内。由专家训练的 TreeStrap ($\alpha\beta$) 达到了与 TD-Leaf 大致相同的水平, 而不需要过滤对手糟糕的移动以及合适的开局等技巧。更重要的是, TreeStrap 从自我对弈中学习的效率更高。虽然专家训练版本的 TreeStrap 仍然比自我对弈的版本趋于更高的排名, 但两者的差异缩小到 ≈ 150 Elo, 打破了自我对弈在国际象棋不起作用的神话。有传闻说 TreeStrap 能够达到 ≈ 2800 Elo (见 Veness 的个人通信)。

以下观察结果展现了 TreeStrap 成功的关键因素: 在前瞻搜索的过程中, 该算法将遍历各种不可能的状态 (使用不理智的棋手会采取的移动)。函数近似必须确保这种不理智状态也被或多或少地正确地评估 (使得它们的值低于正确移动的值), 否则搜索会被价值函数误导。一个额外的因素是, TreeStrap 还能更有效地使用资源: 对于每个树搜索, 它会更新每个内部节点。 [548]

17.2.2.4 为什么国际象棋难以使用强化学习?

我们敢说, 强化学习在国际象棋上的表现并不糟糕。毕竟, 国际象棋已经是数十年来研

⊖ 我们注意到它也可能意味着一个近似错误。

究最积极的游戏之一，强化学习必须与专门针对双人、全信息、零和游戏的树搜索算法进行竞争。此外，大量的精力花在了调整这些搜索算法以使其在国际象棋上表现不俗，以及构建大量的开局库。相比之下，像 TD 学习这样的强化学习方法更通用（例如，能够处理随机性），因此必然不太有效。不过，为什么强化学习不能重复在西洋双陆棋的成功仍然值得深究。

对于基于价值函数的方法，其中一个原因是为国际象棋构造良好的特征并不容易。大多数启发式状态评估函数使用诸如棋子价值（棋子相对于兵的大致价值）、棋子位置价值（棋子在棋盘的每个特定区域上的价值）、编码兵的结构、国王的防御和机动性等特征。这些特征不足以区分具有相似的排列但具有完全不同的价值的状态。Thrun 提到 “knight fork”（见图 17.3）的例子，骑士同时威胁到国王和王后。如果要意识到 knight fork 是危险的（并将其与骑士处于无害的位置区分开来），学习器必须了解棋子的相对位置，并将其推广到绝对位置，这对于基于方形的表示来说是一个相当复杂的任务^①。

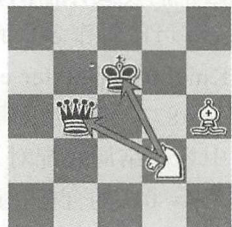


图 17.3 Knight fork：骑士同时威胁到国王和王后

很少有人尝试学习这种模式。Morph 系统的变体 [Gould and Levinson, 1992 ; Levinson and Weber, 2001] 提取局部模式，并通过一个结合 TD 学习的系统来了解其价值和相关性。据说 Morph 比 NeuroChess 强一些，但还是很弱。虽然不是强化学习产品，[Finkelstein and Markovitch, 1998] 的做法也值得一提。他们不使用棋盘模式，而是产生移动模式——在强化学习术语中，我们可以称之为状态特征和状态-动作（或状态-宏）特征。移动模式可以用于过滤或影响移动的选择，尽管没有进行实验来分析对游戏能力的影响。

有趣的是，国际象棋对于状态空间采样法（如 UCT 和其他蒙特卡罗树搜索算法）来说也是很难的，即使在使用启发式评估函数进行增强时也是如此。我们将在 17.2.3.4 节讨论可能的原因。

国际象棋的确定性是另一个使强化学习难以实现的因素。学习器必须通过积极探索从充分多样化的状态空间子集获得经验。适当的探索这个问题基本上是开放的，虽然该问题在进化博弈论中有过抽象的研究 [Fudenberg and Levine, 1998]，但我们知道没有具体到国际象棋的解决办法。此外，确定性使得学习过程对于对手的策略很敏感：如果对手的策略没有足够的多样性（就像在自我对弈中那样），学习将很容易陷入一种平衡，即其中一个玩家的策略有重要的缺陷而另一个玩家却无法利用。

虽然强化学习方法对于控制任务（也就是“玩国际象棋”）还没有竞争力，但它们对于评估来说绝对有用。[Beal and Smith, 1997] 以及后来的 [Droste and Fükurnz, 2008] 使用 TD-Leaf 来学习棋子的子力价值以及棋子-棋盘价值。[Droste and Furnkranz, 2008] 的实验表明，在几个国际象棋程序中，使用学习得到的值的表现优于使用专家值。在非传统象棋的变体（如“自杀象棋”）的实例中效果更明显，这类象棋在 AI 研究中的关注要少得多，精心设计的启发式函数也较少。

17.2.3 围棋

围棋是一种古老的两个玩家的游戏。在 19×19 的棋盘上，玩家轮流放置棋子（见

① 我们注意到这种特殊的情况可以通过静态搜索来解决。

图 17.4)。总体目标是通过占领或包围来攻取比对手更大的领土；完全被包围的棋子会“死亡”并从棋盘上移除^①。由于难度较大，许多程序在更小更简单的 9×9 或 13×13 的棋盘上玩。

到目前为止，最好的围棋程序是大师级别、低于最厉害的人类玩家的水平。由于其难度和受欢迎程度，围棋接管了国际象棋的“大挑战”。尽管围棋是一个双人、全信息、零和游戏，就像国际象棋一样，但是对于传统的基于极值搜索的方法（以及其他方法）要难得多，主要有两个原因。首先，分支因素是巨大的：大多数空白的棋盘位置是合法的。另外，在特殊情况下，人类可以轻松地区分 60 个或更多的半步 [Schraudolph et al, 2001]。目前的树搜索算法还不能接近这个深度。其次，很难为围棋编写好的启发式评估函数。在国际象棋中，子力优势和其他类似度量提供了一个在叶节点中使用仍然足够好的粗略评估。在围棋中，没有这样简单的指标存在 [Müller, 2002]，甚至从“最后”的棋局断定赢家也是不容易的。

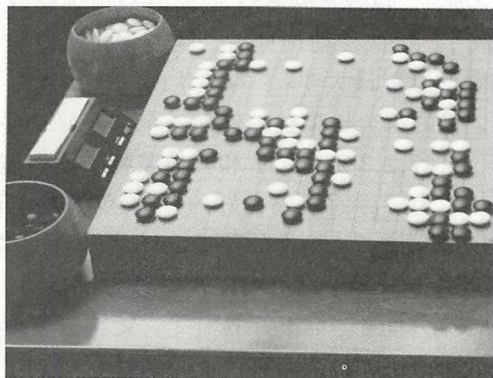


图 17.4 一个 19×19 的围棋棋盘

550

17.2.3.1 特征和模式

在围棋中，评估函数本身不足以指导做出决策，但仍然有助于影响搜索。候选移动的常用特征包括与上一次动作的距离、与边界的距离、捕获的潜力等，有关更多示例，请参见 [Coulom, 2007] 的表 1。这些简单的特征无法捕捉领土的形状，而形状对于确定围棋状态值至关重要。有几种方法使用模式来改进评估，尝试提取相关模式并学习它们的值。

[Schraudolph et al, 2001] 在评估候选移动周围的局部模式的神经网络中使用 TD 学习。其他的技巧还有他们利用了围棋的特殊性质。首先，最终的棋盘状态比单一的赢利 / 失败信息更加翔实：对于每个位置，其对得分的贡献是已知的（ ± 1 ，取决于它所属的领土加上从那里移除的每块棋子分数）。因此，网络可以尝试使用一个输出神经元来为每个位置预测局部奖励。[Dahl, 2001] 也使用神经网络来预测最终的领土，再用一个神经网络来预测团体的安全性（该团体是否会死亡或活到最后）。在复杂决策系统中神经网络与 $\alpha\beta$ -剪枝结合使用。并不清楚到底需要多少学习得到的额外信息才是有帮助的：大多数学习得到的信息可能是多余的。

[Gelly and Silver, 2008 ; Silver et al, 2008] 使用时差法学习所有大小为 3×3 的局部模式的值。他们使用学习的评估函数来指导搜索（见下文）。[Coulom, 2007] 从记录的人类专家的游戏提取了频繁发生的模式。模式的评估是非常规且非常有趣的：每个候选动作被认为是一个模式（和特征）组。竞争获胜的模式组对应于在记录的游戏实际采取的动作。然后使用一般化的 Elo 等级公式来为每个单独的模式和特征赋予评级。

17.2.3.2 Dyna-2

[Silver et al, 2008] 的 Dyna-2 架构以有趣的方式应用 TD 学习。全局评估函数不可避免地平均了许多不同局部情况的值，高估了一些糟糕情况，低估了一些好的情况。另一方面，局部拟合的评估函数在训练的那种情况（那个特定游戏）中更准确，但是并不通用。Dyna-2

① 规则的详细描述可参见 <http://senseis.xmp.net/?RulesOfGoIntroductory>。

[551] 中的基本思想是维持一个存储了许多游戏的平均特征权重并且通过 TD (λ) 学习的“永久记忆”，以及一个基于当前游戏情况修改权重并且也是通过自我对弈 TD(λ) 训练的“瞬时记忆”。每次移动选择后，瞬时记忆被擦除。两个组件的函数近似都是几个基本特征加上每个棋盘位置最大 3×3 的形状模式（导致需要学习数量众多的参数）的线性组合。瞬时和永久记忆的组合比单独的任何一個都要好一些，而 Dyna-2 在较小的棋盘上表现出色。

17.2.3.3 UCT 和其他蒙特卡罗树搜索方法

为了应对缺乏良好的评估函数这一问题，提出了蒙特卡罗 (MC) 采样方法进行启发式棋盘评估：从给定的位置，评估者随机移动来完成游戏并获得最终结果。许多此类快速走子 (rollout) 的平均值可以衡量状态的价值。[Bouzy and Helmstetter, 2003] 给出了围棋中传统 MC 技术的一个很好的概述。蒙特卡罗值估算是强化学习研究领域的研究热点，近期基于多臂赌博机和稀疏采样的先进方法扩展到了游戏树搜索。这为计算机围棋带来了突破。

具有树搜索的传统 MC 评估可以被视为一个包含两部分的搜索树。上部是深度为 d 的完全搜索树（尽管不一定是均匀的深度），其中节点值由极值规则更新；而下部是不完整的（通常甚至不存储在内存中），由最终状态的随机路径组成，节点值是它们的子节点的平均值。最近蒙特卡罗树搜索 (MCTS) 算法去除了尖锐的边界，搜索树可能变得极其不平衡，更新规则从平均值逐渐转换到极值。

第一个这样的算法是 UCT[Kocsis and Szepesvári, 2006]，应用于树的上置信边界，其基于一个用于多臂赌博机的有效的在线学习算法 UCB1[Auer et al, 2002]。对于树的每个节点，UCT 跟踪以前看到的该节点的快速走子的平均收益。如果样本数量低，则样本平均值可能非常不准确，因此 UCT 还计算了收益的高概率置信区间。动作选择是乐观地进行的：令 $n(x)$ 是节点在状态 x 的访问次数（发生相应状态的模拟游戏的数量）， $n(x, a)$ 是选择移动 a 的次数， $m(x, a)$ 是其中的胜利次数。该算法选择具有最高上置信边界的移动

$$Q(x, a) = \frac{m(x, a)}{n(x, a)} + C \sqrt{\frac{\log n(x)}{n(x, a)}} \quad (17.1)$$

[552] 其中 C 是适当大一些的常数，通常通过实验进行调整，并且通常远低于理论（保守）值。实际上，该选择策略仅在至少有 50 ~ 100 次访问状态时使用，否则使用简单的随机移动选择^①。此外，为了节省存储，仅在快速走子的随机部分中的第一批（多个）节点上扩展树。如果 n_i 较小，则 UCT 大致相等地选择每个移动，经过大量试验之后，奖励项可以忽略不计而选择极值移动。

虽然 UCT 的选择公式具有理论基础，但其他公式的尝试也很成功。[Chaslot et al, 2008] 建议 $O(1/\sqrt{n(x, a)})$ 奖励太保守， $O(1/n(x, a))$ 在实践中效果更好。[Chaslot et al, 2009] 给出了一个也可以纳入领域知识的更复杂的启发式公式。[Coulom, 2006] 提出了另一种 MCTS 选择策略，其效果类似（从平均值逐渐转到极值），但选择是概率性的。

虽然 UCT 或其他 MCTS 算法是所有最先进的围棋程序的基础，但是通过启发式信息和使用领域知识引导搜索过程的方法可以大大提高普通 UCT 的下棋水平^②。下面我们概述一些改进方法。

① 从 UCT 移动选择到随机选择的转换也可以逐步完成。[Chaslot et al, 2008] 以为这可以提高效率。

② 计算机围棋服务器上的纯 UCT 玩家的结果请参阅 <http://senseis.xmp.net/?CGOSBasicUCTBots> 和 <http://cgos.boardspace.net/9x9/allTime.html>。其中最强的 Elo 评分为 1645，而最强的无约束程序超过 2500。

偏置树搜索。[Gelly and Silver, 2008] 通过使用启发式函数 $Q^h(x, a)$ 来包含先验知识。当一个节点首次添加到树中时, 初始化为 $Q(x, a) = Q^h(x, a)$ 和 $n(x, a) = n^h(x, a)$ 。数量 n^h 表示在等效经验方面对启发式信息的置信度。该技术可以替代地解释为添加许多虚构的胜利/失败 [Chatriot et al, 2008 ; Chaslot et al, 2008]。逐渐地取消或扩大 [Chaslot et al, 2008 ; Coulom, 2006] 严重削弱了分支因素: 最初, 只把启发式的最佳移动候选添加到每个节点。随着节点访问次数的增加, 会添加更多分支, 得到的候选移动被启发式函数认为是更弱的。

RAVE (快速动作值估算) [Gelly and Silver, 2008] 也称作 all-moves-as-first, 这是一个极其具有围棋特色的启发式方法, 使用玩家在游戏中进行的所有动作并更新其访问次数和其他统计信息。启发式函数处理动作时不关注动作的顺序。这就产生了一个快速但扭曲的价值估计。

偏置蒙特卡罗模拟。令人费解的是, 为什么蒙特卡罗评估有效: 模拟的随机游戏对应于两个非常弱的对手之间的游戏, 不清楚为什么状态评估对更强的对手有用。仿真策略越强结果越好, 这似乎是合理的。这在某种程度上是正确的, 但有两个相反的因素。首先, 为了充分探索, 模拟必须是多样的; 其次, 移动的选择在计算上必须是轻量级的, 因为我们需要每个游戏运行数百万次。因此, 快速走子策略对于 MCTS 方法的成功至关重要。[Gelly et al, 2006] 使用几种有趣的 3×3 的模式, 并选择具有较高概率匹配模式的移动。大多数其他围棋程序使用类似的偏置。

553

[Silver and Tesauro, 2009] 提出, 只要对手“平衡”且他们的错误以某种方式抵消, 快速走子的弱点就不是问题。虽然平均来说对手是一样的强度 (他们是完全相同的), 但随机波动可能使快速走子不平衡。[Silver and Tesauro, 2009] 提出了两种方法来学习较低失衡率的 MC 模拟策略, 并表明它能显著提高性能。

17.2.3.4 为什么 MCTS 在围棋中有效?

UCT 和其他 MCTS 方法的成功使其在多种类型的其他游戏中得到应用, 但它的成功并不能复制, 最著名的就是国际象棋 [Ramanujan et al, 2010]。那么问题就出现了: 为什么 UCT 和其他 MCTS 方法在围棋中运作良好? 从理论的角度来看, [Coquelin and Munos, 2007] 表明, 在最坏的情况下, UCT 比随机搜索执行得更差: 为了找到近似最优策略, 所需样本数量能以深度的嵌套指数函数增长。他们还提出了一种替代的 BAST (用于平滑树的强盗算法), 其具有较为正常的最坏情况。有趣的是, BAST 较好的最坏情况并不会导致其在围棋中有更好的表现。可能的原因是不保守的更新策略在经验上更好 [Chaslot et al, 2008], 而 BAST 比 UCT 更加保守。

策略偏置与价值偏置。根据 [Szepesvári, 2010, 个人通信], MCTS 的成功可以 (至少部分地) 归因于这样一个事实, 即与 TD 之类的基于价值函数的方法不同。价值估计的函数近似对搜索产生了“价值偏置”。另一方面, 树搜索方法中使用的启发式信息在策略空间中引导了搜索, 形成了“策略偏置”。两种偏置具有不同的优缺点 (也可以结合起来, 如 Dyna-2), 对于围棋来说, 似乎策略偏置更容易做好。

窄路径、浅陷阱。UCT 估计一个状态的价值时使用结果的平均值。平均值集中在极值附近, 但可能需要很长时间才收敛。[Coquelin and Munos, 2007] 想象一个假设的情况, 黑方有一条“狭窄的胜利之路”: 如果他正确地执行某种移动顺序就赢了, 但如果他即使仅犯了一个错误都会输。状态的极值就是“赢”, 但平均值将更接近于“输”, 除非一半的权重集中在获胜分支上。然而, 该算法必须遍历完整的搜索树才能发现这一点。

以相同的方式，平均操作可以被相反的情况所欺骗，即对手有一个赢的策略而任何其他策略都不行。除非我们让 MCTS 花很长一段时间来弄清楚情况，否则就会陷入困境，认为当前状态是安全的。如果对手从给定状态找到了较短的获胜策略，那么这就是极其不利的：然后即使是一个浅的极值搜索也可以找到获胜策略并击败 MCTS。[Ramanujan et al, 2010] 将这类情况称为“浅陷阱”，并认为国际象棋中的糟糕移动往往会导致浅陷阱，使国际象棋成为 MCTS 方法的一个难题。

在围棋中，窄路径和浅陷阱似乎不太普遍：糟糕的动作会导致许多步以后的失败而极值搜索方法不能前瞻太远，所以 MCTS 可以占据上风。然而，在围棋中也存在陷阱。[Chaslot et al, 2009] 确定了一种名为“点杀”（nakade）的情况，这是一种被包围的团体有单一的较大的内部封闭空间的情况，如果对手正常玩，玩家将无法建立两只眼。因此，该团体已经死了，但基准的蒙特卡罗模拟器有时候却估计它有高存活概率。[Chaslot et al, 2009] 使用局部极值搜索来单独识别和处理 nakade。

17.2.4 俄罗斯方块

俄罗斯方块是 1984 年由 Alexey Pajitnov 创造的在 10×20 的棋盘上玩的电脑游戏（见图 17.5）。方块一个接一个地下落，每个新的方块一律随机产生。玩家可以控制当前方块掉落的位置也可以进行旋转。窗口被无法消除的方块填满时游戏结束，但填满的行会被清除。得分通过清除行（和其他东西）获得，游戏的目标是最大化得分。在一个广泛应用于 AI 研究 [Bertsekas and Tsitsiklis, 1996] 的更抽象的变体中，方块不会掉落，玩家只需要给出当前方块的旋转以及方块下落的列。此外，得分是扁平的，即每条清除的行都是获得 +1 的奖励^①。存在许多不同的棋盘规模、形状集、形状分布的变体。

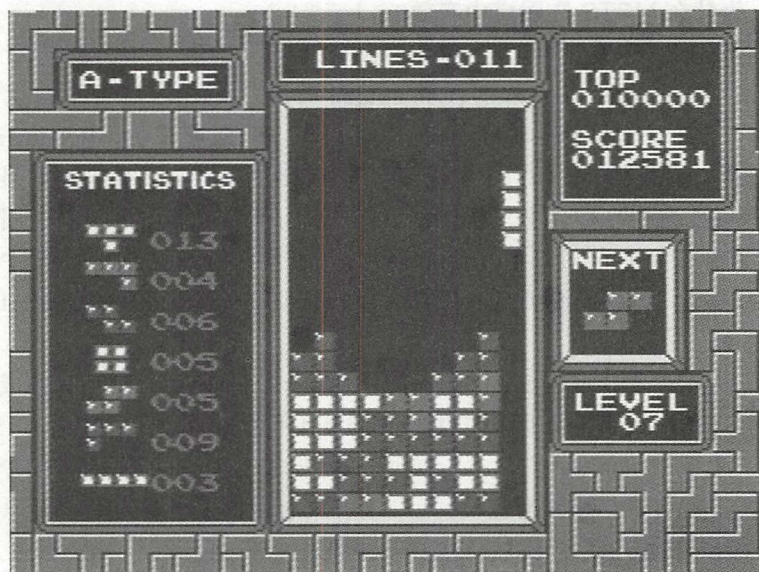


图 17.5 任天堂娱乐系统（NES）的俄罗斯方块的屏幕截图。左侧列列出了七种可能的俄罗斯方块。每个方块的标准名称依次为 T, J, Z, O, S, L, I

① 原始的俄罗斯方块的详细规则参见 http://www.colinfahey.com/tetris/tetris_en.html 的第 5 章；对于抽象版本的规范参见 [Bertsekas and Tsitsiklis, 1996]。

作为一个具有随机性的完全可视的单人游戏，俄罗斯方块非常适合 MDP 框架，而大部分强化学习研究都集中在其中。有另外两个因素助其流行：其实施的简单及其最佳控制的复杂性。即使是近似，即使预先知道方块出现的序列，方块的最佳放置也是 NP 难的问题。因此，俄罗斯方块成为测试和基准测试强化学习算法最受欢迎的游戏。

除了使俄罗斯方块成为强化学习的良好基准的属性外，它还有一个不幸的属性——策略价值的方差是巨大的，与其平均值相当。此外，一个策略越好，评估它的时间就越长（我们必须玩完游戏）。这使得比较顶层算法或调查细微变化的影响变得非常困难。因此，[Szita and Szepesvári, 2010] 提出使用仅包含“S”和“Z”形方块的变体 SZ-Tetris。他们认为，这种变体是俄罗斯方块的“硬核”，也就是说，它保留（甚至放大）俄罗斯方块的大部分复杂性，最大分数（和方差）要低得多，因此这似乎是一个更好的实验测试平台。

俄罗斯方块也是特殊的，因为它是价值函数的函数近似无法得到较好效果的少数应用之一。许多价值函数近似的强化学习方法都有偏离或仅具有弱的性能界限。然而，理论上的负面结果（或缺乏正面结果）经常由于过于保守而被忽视，因为价值函数近似在实践中通常表现良好。俄罗斯方块是一个有用的提醒，即使对于实际感兴趣的问题，理论性能界限也不应如此松散。

[Bertsekas and Tsitsiklis, 1996] 是在俄罗斯方块中首先尝试强化学习的人之一。状态被表示为由独立的列高（10 个特征）、列高差值（9 个特征）、最大列高和孔的数目组成的 21 个元素的特征向量。这种具有线性函数近似的表示作为大多数后续方法的基础^①，也为比较“基于价值函数”和“基于偏置函数”的方法提供了极好的机会。在这两组中，特征函数 $V(x) = \sum_{i=1}^k w_i \phi_i(x)$ 的线性组合用于决策，但在第一组方法中， $V(x)$ 尝试近似最优价值函数 $V^*(x)$ ，而在第二组中，没有确保 $V(x) \approx V^*(x)$ 的尝试，这使得参数优化更不受约束。偏好功能也可以视为直接策略搜索方法的特殊情况。

556

17.2.4.1 基于价值函数的方法

为了学习权重 w_i ，[Bertsekas and Tsitsiklis, 1996] 应用了可以认为是 TD(λ) 的规划版本的 λ -策略迭代，并生成值迭代和策略迭代。他们报告的最佳性能是 3200 点，但继续训练性能会变差。[Lagoudakis et al, 2002] 尝试了最小二乘策略的迭代；[Farias and van Roy, 2006] 将近似线性规划应用于生成样本的迭代过程；[Kakade, 2001] 采用自然策略梯度调整权值。这些方法的性能在 3000 ~ 7000 的同一数量级内。

17.2.4.2 基于偏好函数的方法

[Böhm et al, 2004] 利用进化算法调整权值向量，使用两点交叉和高斯噪声作为突变算子，以及一个不影响权值向量拟合的随机权值尺度改变算子，但是改变了它如何对未来的遗传操作做出反应。[Szita and Lörincz, 2006a] 应用了交叉熵法（CEM），其维持了参数空间的高斯分布，并且适应分布的均值和方差以便最大化从分布中采样的权值向量的性能，同时保持在足够探索的水平。[Thiery and Scherrer, 2009] 使用额外的特征函数后，其结果有所改进，达到 3500 万点。虽然 CEM 从一个独立的高斯（即其联合分布是轴对齐的高斯）绘制每个向量分量，但是 CMA-ES 方法（协方差矩阵适应的进化策略）允许使用一般协方差矩阵，代价是增加算法的参数数量。[Boumaza, 2009] 将 CMA-ES 应用于俄罗斯方块，达到了类似于 [Thiery and Scherrer, 2009] 的成果。

① 虽然设计和尝试了许多其他特征；[Thiery and Scherrer, 2009] 的表 1 总结了这些内容。

17.2.4.3 偏好函数与近似价值函数以及特征在俄罗斯方块中的作用

给定近似价值函数 V , $V(x)$ 大致是从 x 得到的总的奖励。偏好函数没有这个额外的意义, 只有比较意义: $V(x) > V(y)$ 意味着在算法中 x 优先于 y (可能是因为它可以从 x 获得更多的奖励)。无限多的偏好函数引出与给定价值函数相同的策略, 因为由偏好函数引出的策略对于尺度改变或任何其他单调递增的变换来说是不变的。贝尔曼最优方程 V^* 的确切解也是一个最优偏好函数, 因此解决第一个任务也就解决了另一个。在函数近似的情况下, 这不再

557 是正确的, 在最小化近似贝尔曼误差方面相当不错的价值函数可能在作为偏好函数时非常糟糕 (它可能会以错误的顺序排列几乎所有的动作, 虽然仍具有较小的贝尔曼残差)。此外, 近似贝尔曼方程的解也没有意义。事实上, 它可以是距离 V^* 任意远的。通常情况下, 这种情况只发生在人为的反例中, 但俄罗斯方块的传统特征表示却很弱, 这表明这个问题也受到这种现象的影响。[Szita and Szepesvári, 2010] 避免问题的一种方法是改变特征函数。然而, 是否可以统一偏好函数 (直接性能优化) 和近似价值函数 (近似贝尔曼方程允许自举: 从 $V(x_{t+1})$ 学习) 的优点, 这是一个有趣的开放性问题。

17.2.4.4 俄罗斯方块中的强化学习算法有多好?

在某些方面, 俄罗斯方块的强化学习学习器远远超出了人类的能力: [Thiery and Scherrer, 2009] 的学习器平均可以清除 3500 万行。以 1 方块/秒的速度计算 (对于一个人类玩家来说相当不错), 这将需要 2.77 年的游戏时间。然而, 观察 AI 的玩法表明, 它们会采取人类尽量避免的不直观的 (可能是危险的) 动作。SZ-Tetris 的结果目前也支持人类的优越性: CEM 训练的控制器达到 133 分, 而一个手动编码控制器能达到 182 分, 比优秀的人类玩家稍弱 [Szita and Szepesvári, 2010]。

17.2.5 即时战略游戏

即时战略 (Real-Time Strategy, RTS) 类型的游戏是一种战争模拟。在典型的 RTS (见图 17.6) 中, 玩家需要收集资源, 将资源用于建设军事基地、进行技术开发和训练军队、摧毁对手的基地、防范对手的攻击。玩家的任务包括确保充足的资源收入、经济扩张和军事力量之间的资源配置平衡、建筑物的战略布置和防御、探险、进攻规划和战斗中单位的战术管理。由于 RTS 游戏的复杂性, 强化学习方法通常选择一个或几个子任务来学习, 而其他的依赖于默认启发式方法。子任务列表可以在 [Buro et al, 2007; Laursen and Nielsen, 2005; Ponsen et al, 2010] 之中找到。RTS 游戏有一个在前面列出的任何游戏中都不存在的独特之处——需要处理并行任务。这是一个巨大的挑战, 因为并行任务可以在任务间交互且可以跨越不同的时间尺度。

558 大多数实现使用的是魔兽争霸和星际争霸系列、开源 RTS 系列 Freecraft / Wargus / Stratagus / Battle of Survival/ BosWar[⊖], 或专门用于促进研究而开发的 ORTS 游戏引擎 [Buroa and Furtak, 2003]。各种各样的任务和各种不同的 RTS 游戏伴随着多种可选的强化学习方法。

17.2.5.1 动态脚本

动态脚本 (DS) 最初应用于角色扮演游戏的战术作战 [Spronck et al, 2006], 但随后就应用到了 RTS 中。动态脚本通过从规则池中随机抽取 K 个规则来组合策略。选择规则的概率与其权重成比例, 最初所有的权值是相等的。抽取的策略在游戏中进行测试, 其规则将

⊖ 这个包含很多名字的家族内部的关联参见 <http://en.wikipedia.org/wiki/Stratagus#History>。

根据结果得到加强：如果策略能获胜，权重就会增加，否则会下降。其他规则的权重也会调整以保持权值的总量不变，并且可以规定最小 / 最大权值以维持足够的探索。[Ponsen and Spronck, 2004] 在 Wargus 中通过为游戏的每个抽象状态学习单独的脚本部署了 DS。状态由一系列学习器的基地里的建筑物来定义（这在很大程度上决定了可用动作的集合）。每个状态脚本都是通过结合最终结果和离开当前状态时的启发信息的评估来单独地进行奖励。脚本的规则属于以下四类之一，建造、研究、资源或战斗。不同类别的规则可以并行运行。动态脚本比默认的 AI 和大多数手动编码的 AI（尽管不是接近最优的策略，即“骑士冲锋”）更优秀。[Ponsen and Spronck, 2004] 的适应函数使用简单的材质强度评估（类似于在国际象棋中某个程序使用的）。[Kerbusch, 2005] 已经表明，TD 学习可以比手动编码更好地调整单位的相对优势，更新的评估函数可以提高动态脚本的游戏强度。[Kok, 2008] 使用隐式状态表示，每个规则的前提条件决定了规则是否适用。此外，可用的动作数量要低得多，通常大约是五个。通过这些修改，他在使用 ϵ 贪婪探索及考虑到规则的固定排序的修改后，普通 DS 和蒙特卡罗控制（相当于 TD(1)）两种方法都达到了良好的效果。

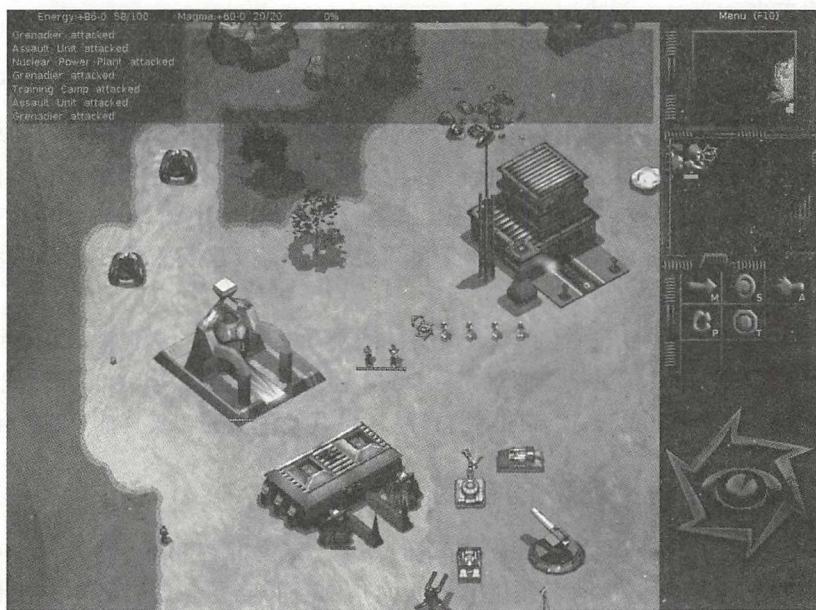


图 17.6 一个开源的即时战略游戏 BosWars 的屏幕截图

17.2.5.2 演化学习

[Ponsen and Spronck, 2004] 也在 Wargus 上尝试了演化学习，每个建筑的状态有四个基因，对应于四个动作类别。在后续论文中，[Ponsen et al, 2005] 通过预演动作集及通过选择每个状态联合起来效果较好的几个四元组动作来获得改进的结果。

EvolutionChamber 是最近尚未发表的用于学习 Starcraft 的开局动作的方法[⊖]。它使用游戏的一个精确的模型来尽可能快地达到预定的目标条件（如创建至少一个类型的 K 个单位）。EvolutionChamber 假设对手没有互动，并且需要一个目标条件作为输入，因此目前的形式只

⊖ 详细说明参见 <http://brandy.com/blog/2010/11/using-genetic-algorithms-to-find-starcraft-2-build-orders/>，源代码参见 <http://code.google.com/p/evolutionchamber/>。我们注意到，尽管算法没有通过学术期刊或会议记录发表，其算法和结果都是公开的并且已经得到该领域的专家验证。

适用于开局。然而，一个巨大的优势是，它能产生甚至对于人类玩家来说可以直接应用到星际争霸的开局战略，并且至少已经产生了一个以前未知的、也被专业玩家认为非常有效的开局（“7 螳螂 RUSH”，五分钟内得到了七个重型装甲远程单位）[⊖]。

17.2.5.3 基于案例的学习

在基于案例的推理中，学习器存储几种相关的情况（案例），并告知如何处理这些情况的策略。相关案例可以手动 [Szczepański and Aamodt, 2009] 或自动（脱机或在线）地添加到案例库。当学习器面对新的情况时，它会查找存储的最相似的案例并根据它们做出决策，并且（当涉及学习时）它们的统计信息将根据强化来更新。[Aha et al, 2005] 使用的包含动作集和评估函数的抽象状态与 [Ponsen and Spronck, 2004] 使用的是一致的，案例的距离由八维的特征向量确定。如果一个案例与所有存储的案例显著不同，则将其添加到池中，否则更新现有案例的奖励信息。所得到的算法胜过了 [Ponsen and Spronck, 2004] 的强化 AI，并且能够应对一些未知的对手 [Molineaux et al, 2005]。[Weber and Mateas, 2009] 引入了类似于“编辑距离”的改进的距离度量：它们将两种情况间的距离定义为从一种情况转到另一种所需的资源的大小。

17.2.5.4 在即时战略中强化学习方法有多好？

列出的方法都是处理高层决策，而将战术作战或基础安排等下层任务委派给默认策略。不顾它们之间可能的相互作用肯定会损害其表现，例如，如果要求士兵的进攻协调得很好而且能够迅速杀死对手，那么他们就少有合格的。但是，即使不计较这一点，目前的方法仍有明显的缺点，例如处理并行决策的能力有限（解决方案参见 [Marthi et al, 2005] 或有限的探索能力。例如，[Ponsen and Spronck, 2004] 报告说，有高级建筑的游戏后期的状态在训练中很少出现，导致其策略较弱。

RTS 的最先进的电脑玩家远远低于最厉害的人类玩家的水平，例如，2010 年 AIIDE 星际争霸大赛冠军 Overmind 与退役职业玩家 = DoGo = 之间的比赛。如今的即时战略顶级 AI，例如“星际争霸”比赛和 ORTS 比赛的参赛作品，没有使用上述（或其他）的强化技术。

17.3 强化学习应用到游戏的挑战

前一节深入探讨了几种游戏及其解决方案的技术。选择的各自体现了一些有趣的强化学习问题，并且它们一起展示了强化学习所面临的各类问题。这个名单并不代表一种标准。在本节中，我们列出了将强化学习应用于游戏中面临的重要挑战。

17.3.1 表示的设计

正如在 17.1 节中指出的那样，合适的表示这个问题在强化学习的所有应用中都起核心作用。我们对游戏通常有丰富的领域知识，这有助于表示的设计。领域知识可能有多种形式：

- 规则、游戏模式的知识，了解哪些信息是相关的。
- 来自人类专家的游戏信息。这可能以建议、经验法则、游戏的相关特征等形式出现。
- 游戏记录数据库、开局库、残局库。
- 丰富的评估反馈。在多数情况下，学习器获得的是评估信息而不仅仅是奖励信号（但

⊖ 请参阅 http://www.teamliquid.net/forum/viewmessage.php?topic_id=160231 的论坛讨论。

比完全监督要少), 就像根据一些子目标来分解奖励一样。例如, 在一个作战队伍中, 除了全局奖励(胜利/失败)之外, 每个成员都会获得个人反馈(剩余的生命值、承受/抵抗的伤害总量)。

因此, 在游戏应用中, 我们可以找到各种方法来将领域知识纳入表示, 以及尝试用尽可能少的领域知识获取表示的方法。

17.3.1.1 特征提取、抽象

从原始的感官信息中提取相关信息是获取表示的重要组成部分。原始观察通常被假定为某种预处理形式, 例如, 从屏幕捕获中提取状态信息通常不作为表示设计问题的一部分(并且委托给机器学习的其他领域, 如机器视觉)。有两个有趣的例外: 1) Pac-Man 比赛^①的参赛者需要使用 15 帧/秒的视频捕获作为输入来编写控制器, 但即使使用非常精确的游戏模式, 这也是一个非常不简单的任务; 2) [Naddaf, 2010] 尝试从他们的存储仓库(包括视频信息)中学习 Atari 的 2600 场游戏^②。

输入的特征映射扮演了两个角色(不是彼此独立的): 它们抽象出不太相关的信息, 并将相似的情况映射到相似的特征向量以促进泛化。“相似性”的含义在很大程度上取决于由特征向量构成的架构。多数情况下, 架构旨在近似价值函数, 尽管也可以利用基于模型或结构化(关系, 因子分解)的强化学习方法。函数近似可以是线性的或非线性的, 在后一种情况下, 神经网络是最常见的选择。

下面列出了几种可能的表示类型。它们的使用(无论是无模型函数近似还是基于模型的架构)都在本书的其他部分进行了详细的介绍, 所以这里只给出一个说明: 大多数算法在使用了函数近似时都不能保证收敛, 但却广泛应用, 并且没有报告任何问题。我们注意到, 部分原因绝对是“呈现的偏差”: 如果应用程序因参数灵敏度/发散/收敛慢/糟糕的特征而不起作用, 那么其发布的概率很低。

表格表示。对于具有离散状态的游戏, 简单的特征映射为每个状态提供唯一的标识符, 对应于“表格表示”。基于动态规划的方法的性能取决于状态空间的大小, 这通常使表格表示变得难以处理。然而, 诸如 UCT 的状态空间采样方法仍然易于处理, 因为它们避免了每个状态的值的存储。它们通过增加动作前的决策时间来在存储的值的缺乏和泛化的缺乏之间进行折衷。例如, 请参阅围棋的应用程序(见 17.2.3 节) Settlers of Catan[Szita et al, 2010] 或一般游戏[Björnsson and Finnsson, 2009]。

专家特征。游戏的场景通常是由专家认为相关的几个数值来刻画的。这些特征的例子有国际象棋的子力强度、围棋中的自由度, 或者俄罗斯方块的孔的数量。使用二进制特征也是非常常见的, 例如国际象棋或围棋中的模式。

关系表示。关系的 [Džeroski et al, 2001]、面向对象的 [Diuk et al, 2008] 和指示的 [Ponsen et al, 2006] 表示是相似的, 它们假设游戏场景可以通过对象之间的几个相关的关系来刻画(从技术上来说, 是一种专家特征的特殊情况, 每个关系仅涉及几个对象)。这些表示可以直接用于价值函数近似 [Szita 和 Lörincz, 2006b] 或用于一些结构化的强化学习算法

① 见 <http://cswww.essex.ac.uk/staff/sml/pacman/PacManContest.html>。比赛由西蒙·卢卡斯(Simon Lucas)进行协调, 并且在各种演化计算和游戏 AI 会议(WCCI、CEC 和 GIG)上都组织过。

② Atari 2600 系统是简单和复杂的一个奇妙组合: 其存储器(包含视频存储器)为 1024 位, 还包含视频输出(因为电子束在绘制时必须计算视频)。这些信息量处于可管理的边缘, 是计算机状态的马尔可夫表示。另一方面, Atari 2600 是足够复杂的(和受欢迎的), 这让它发布了近 600 种不同的游戏。

(见第 8 章和第 9 章)。

状态聚合、聚类。虽然在某些情况下可以直接枚举并聚合基态 [Gilpin et al, 2007], 但是在一些先前创建的特征空间中执行聚类是更常见的。聚类的中心可以用作基于案例推理的案例, 也可以用作状态空间的高级离散化。

组合特征。联合特征需要多个 (二进制) 特征一起出现, 而更一般地, 我们可以通过任何逻辑公式来实现特征的组合。组合特征是在输入中引入非线性的一种简单的方法, 因此它们对于线性函数近似的应用特别有用。组合特征也是自动表示搜索技术的常见主题 (见下文)。

17.3.1.2 自动特征构造

563 神经网络的隐藏层定义了输入空间的特征映射, 因此可认为使用神经网络的方法是隐式特征构造的方法。[Tesauro, 1995] 报告称, TD-Gammon 学习了有趣的模式, 与西洋双陆棋的典型情况相对应。[Moriarty and Miikkulainen, 1995] 使用人工进化来为 Othello 位置评估开发神经网络, 并设法发现先进的 Othello 策略。

[Gould and Levinson, 1992] 的 Morph 系统提取和概括了国际象棋位置的模式, 就像 [Levinson and Weber, 2001] 做的一样。两种方法都是通过 TD(λ) 来学习模式的权重。[Finkelstein and Markovitch, 1998] 修改了搜索状态-动作模式的方法。其差异类似于价值函数 $V(x)$ 和 $Q(x, a)$ 的差异: 后者不需要用于决策的前瞻。

特征构造的一般方法是将原子特征组合成更大的联合或更复杂的逻辑公式。[Fawcett and Utgoff, 1992] 的 Zenith 系统使用分解、抽象、回归和特殊运算符从一组 (一阶谓词演算的) 原子公式中为 Othello 构建特征。通过添加有助于区分好的与坏的结果的功能迭代地创建特征。这些特征被裁剪并用于线性评估函数。[Utgoff and Precup, 1998] 的一个类似系统通过更简单的布尔谓词的连接来为跳棋玩家 (Checkers) 构建新的特征。[Buro, 1998] 使用类似的系统来为 Othello 构建组合特征。关于游戏特征构建方法的一个很好的综述参见 [Utgoff, 2001]。关于研究联合特征的最新成果包括 [Shapiro et al, 2003] 的 Diplomacy、[Szita and Lörincz, 2006b] 的 Ms. Pac-Man、[Sturtevant and White, 2007] 的 Hearts。

[Gilpin and Sandholm, 2007] 研究了扑克的抽象 (更普遍的是双人的不完全信息博弈), 并提供了一种保持游戏平衡的状态聚合算法, 即保留关于最优解的相关信息。一般来说, 为了保证任务的大小可行, 必须牺牲一些信息。[Gilpin et al, 2007] 提出了另一种将信息状态分组到固定数目的簇中的自动抽象方法, 使得簇内的状态的未来奖励分配大致相同。[Ponsen et al, 2010] 概述了 (无损和有损的) 即时战略游戏特有的抽象运算符。

一般来说, 在没有任何领域知识的情况下学会表示似乎很难 [Bartók et al, 2010], 但是有一些方法试图 “最小化” 所需领域知识的总量。可以在通用游戏玩法 (general gameplay) 文献中找到例子, 例如 [Sharma et al, 2009; Günther, 2008; Kuhlmann, 2010]。这些方法寻找在大多数游戏中能发现的常规属性和不变量: 空间/时间类型的变量、计数类型变量等。

17.3.2 探索

564 探索是推动学习器达成最优解的核心问题。在大多数游戏应用程序中, 算法期待看到游戏的动态性或由对手进行探索 (针对各种对手进行训练), 或者仅仅是希望一切顺利。我们将在 17.3.3 节中概述挑选对手的相关准则, 并且这里主要集中探讨学习器在产生多样性时自

身的角色。

Boltzmann 和 ϵ 贪婪动作选择是最常见的探索形式。我们从有限 MDP 理论中知道，有更多的有效探索方法存在（见第 6 章），但尚未知道如何将 these 方法扩展到具有函数近似的无模型强化学习（其涵盖强化学习的大部分游戏应用程序）。UCT 和其他蒙特卡罗的树搜索算法以更具理论基础的方式处理探索，维持置信区间意味着结果的不确定性，并应用“乐观地面对不确定性”的原则。当游戏的生成模型可用（许多游戏都是这种情况）时，即使是传统的规划方法也可以模拟探索。例如，在 TD-Leaf 和 TreeStrap 中，有限的前瞻搜索就能确定最有希望的方向。

演化强化学习方法以基于群体的方式探索参数空间。遗传算子（如突变和交叉）确保学习器充分尝试各种不同的策略。RSPSA[Kocsis et al, 2006] 也在参数空间中进行探索，但是使用的是局部搜索而不是维持一定数量的学习器群体。

17.3.3 训练数据的来源

对于具有两个或更多玩家的游戏，学习器的环境取决于其他玩家。学习器的目标可以是找到一个纳什均衡、一个针对固定对手的最佳反应，或是对阵一系列强大的对手时表现良好。然而，如果对手比学习器的实际水平要强得多，学习过程可能非常低效或完全卡住：初级的学习器尝试的所有策略都将有非常高的概率失败，因此强化信号将一律是负的——没有提供任何有用的方向。另一方面，对手必须具有足够的多样性，以防止学习器收敛到局部最优。因此，选择训练对手是一个非常重要的问题。

17.3.3.1 自我对弈

自我对弈是目前最受欢迎的训练方法。根据经验，如果学习器与对手的水平大致相同，训练是最快的，并且这是完全适用于自我对弈的。受欢迎的第二个原因是，没有必要实现或访问具有大致相同实力的不同学习器。但是，自我对弈也有几个缺点。理论上来说，它甚至不能保证收敛 [Bowling, 2004]，尽管没有人提到这在实践中会造成问题。自我对弈的主要问题是单个对手不能提供足够的探索。缺乏探索可能会导致像 [Davidson, 2002] 的 6.1.1 节所述的情况，扑克学习器被自我实现预言（self-fulfilling prophecy）[⊖]所困扰：假定一个状态（不正确地）估计为糟糕的。因此，学习器弃牌（fold）并输掉了游戏，所以状态的估计值会进一步下降。

[565]

17.3.3.2 辅导

作为自我对弈的替代方案，学习器可能会由一组实力递增的不同对手来辅导（tutoring）。对于更受欢迎的游戏（如国际象棋和围棋），这样的一系列对手可以在基于玩家实力来匹配玩家的游戏服务器上找到。然而，这样的游戏池不是每个游戏都有的。即使有，也通常要慢得多，这就使可玩的游戏数量变少了。从积极的角度来说，通过辅导训练的学习器通常比通过自我对弈训练的更强，例如 [Baxter et al, 2001; Veness et al, 2009]。

17.3.3.3 其他训练策略

[Epstein, 1994] 提出了一个“课程和实践”的设置，其（资源昂贵的）辅导游戏的过程之后是自我对弈的过程。[Thrun, 1995] 通过自我对弈训练了 NeuroChess，但是为了强化探索，每个游戏都是通过从游戏数据库随机选出的某个游戏的几个步骤进行初始化的。这样，

⊖ 自我实现预言是指使自己的预期成真的预言。——译者注

学习器可以从一个相对随机的位置开始自我对弈。

从游戏数据库的观察中学习是一种廉价的学习方式，但通常不会带来好的结果：学习器只能体验到良好的移动（这是在数据库游戏中被采纳的），所以它不曾学到更坏的动作的值以及该动作表现不好的场景。

17.3.4 处理缺失的信息

学习器可能必须面对经典游戏和电脑游戏中的信息缺失，例如扑克中的隐藏牌（hidden card）（图 17.7）、第一人称射击游戏的部分可观察或即时战略中的“战争迷雾”（fog of war）。在这里，我们只考虑“真正缺失”的信息，即通过使用更好的特征表示仍然无法获得的信息。此外，关于对手动作的信息缺失在 17.3.5 节中单独讨论。

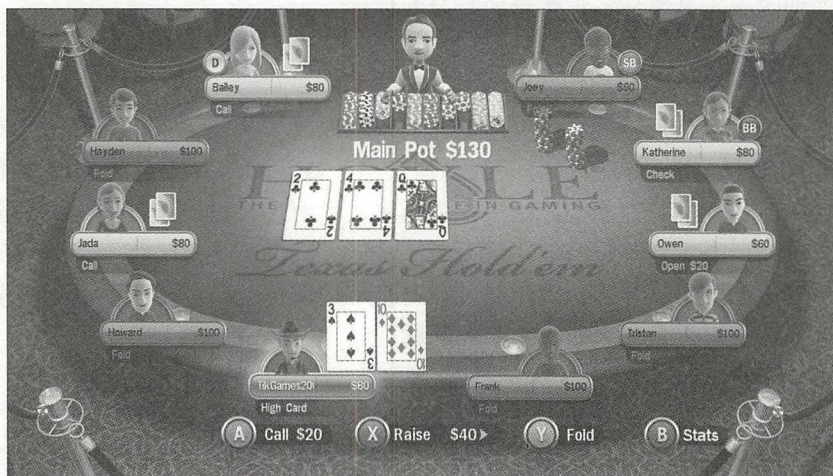


图 17.7 一个正在进行的德州扑克游戏。来自 Xbox 360 上的 Hoyle's Texas Hold Em 屏幕截图

经常使用的方法之一是忽略该问题，并根据当前观察来学习应对策略。虽然类似 TD 的方法可能在局部可观测的情况下发散，但有报告称具有较大 λ 值（包括蒙特卡罗估计）的 $TD(\lambda)$ 是相对稳定的。包括动态脚本 [Spronck et al, 2006]、交叉熵策略搜索 [Szita and Lörincz, 2006b] 和进化策略搜索 [Koza, 1992] 在内的直接策略搜索方法不受局部可观测性的影响。

然而，蒙特卡罗采样在对抗搜索中却有问题：在对未观察到的量进行采样后，学习器认为它们是可观察的，并且还假定对手可以观察到一切。因此，决策将忽略“信息的价值”，即获得额外的信息（或让信息对手隐藏）的动作会优先选择。[Ginsberg, 2002] 引入了一种对“信息集”（无法通过学习器已有信息来区分的状态集）的极值搜索的变体。他们在桥牌程序 GIB 中应用了一个简化的版本。[Amit and Markovitch, 2006] 使用 MC 采样，但对于每位玩家来说，他们未知的信息被掩盖了。该技术应用到了桥牌的出价中用以改进效果。

“反事实遗憾”（Counter Factual Regret, CFR）的概念 [Zinkevich et al, 2008] 与 MCTS 类似，但它在价值计算时抵消了对手的影响，适用于有隐藏信息的游戏。CFR 与状态空间的复杂抽象的组合是目前扑克中最先进的技术 [Risk and Szafron, 2010 ; Schnizlein et al, 2009]^①。

① 参见 [Rubin and Watson, 2011] 对计算机扑克的全面评述。

17.3.5 对手建模

对手的动作是一种隐藏的信息。对手模型试图根据对过去的观察（可能来自许多以前的游戏）来预测对手的动作。对手建模是多人强化学习系统的一个重要增强，尽管它通常不被形式化为一个任务本身。在双人零和游戏中，使用极值最优策略或在部分可观察的情况下，纳什均衡策略即使在最坏的情况下也能保证良好的表现——但实际上，如果对手不够完美，那么利用他们的弱点可以做到更好。扑克游戏系统 Vexbot [Billings et al, 2006] 学习了一个对手模型并计算出最佳响应策略。根据 [Lorenz, 2004] 的说法，即使在完全信息游戏中，对手的建模也是有用的。他观察到，在对抗强大但不完美的玩家的比赛中，极值最优策略没有一个整体上的良好表现：它太保守，无法利用其他玩家的失误。[Davidson, 2002] 用 “Probimax”（在早期文献中也称为 Expectimax）替代了极值搜索，它通过概率分布来模拟对手的策略，并根据这种分布来选择对手在树中的移动。他展示了 Probimax 在国际象棋上的使用：在一个理论上有所损失的状态下，极值认为每个候选移动都是同样糟糕的（所有候选都会导致损失）。另一方面，Probimax 能够区分不同移动，并选择对手最有可能犯错的那个移动。

567

17.4 在游戏中使用强化学习

到目前为止，本章重点关注了强化学习的视角：游戏中特有的算法问题和各种挑战。在本节中，我们来看看强化学习与游戏间关系的另一面：强化学习在游戏中如何生效。当目标是找到一个强大的 AI 玩家时，这两个观点或多或少是一致的。然而，游戏中的强化学习可以在更多的角色中起作用，这在现代视频游戏中是最明显的。在本节中，我们将介绍强化学习如何在游戏开发和游戏设计中使用。

17.4.1 最具娱乐性的对手

与 AI 研究中的游戏相反，商业视频游戏的目的是保持玩家的娱乐性 [Scott, 2002 ; Gilgenbach, 2006]。这个任务的一部分是将难度设定在正确的水平上，既不太难也不容易，奖励获得了难得的胜利的玩家。在这方面，一个理想的 AI 对手应该比玩家的最佳状态稍微弱一些。然而，同样重要的是游戏学习器以令人信服的方式输掉。根据 [Gilgenbach, 2006] 的说法，学习器需要创造一种尽管它的动作是智慧的并且尽最大努力追求胜利但还是失败的错觉。当然，“乐趣”“挑战”和“智慧”都不是很清晰的概念，允许许多不同的解释。下面我们回顾一些适用于强化学习的内容。

568

在大量的游戏中，使计算机足够聪明是一个挑战，但如同乒乓球游戏展示的例子一样，使其足够弱也可能同样如此。乒乓球游戏是第一个有电脑对手的视频游戏。最优策略是不重要的（球拍应该跟随球的 y 坐标），但显然没有多少人玩乒乓球会打败最优策略。想象一下如果现在的学习器完全可以在 45% 的时间内做得非常完美而其余时间不做任何事情——与这个学习器的比分或许能比较接近，但可能还是没有太多的乐趣。[McGlinchey, 2003] 试图从人类玩家的记录中提取可信的策略。对人类动作建模是一种流行的方法，但不在本章的范围之内，可参见 [Hartley et al, 2005] 的概述。

17.4.1.1 难度缩放

难度缩放（difficulty scaling）技术会自动调整游戏的难度。它们采取强有力的（可能是适应性的）策略，如有必要，可以将其水平调整到玩家的水平。

[Spronck et al, 2004] 修改了动态脚本, 在三个修改中最成功的一个上面, 他们排除了使策略太强的规则。他们保持了一个胜利降低且失败增加的权值阈值 W_{max} , 如果规则的权值超过 W_{max} , 则不能选择。他们在 RPG 游戏 Baldur's Gate 的简化模拟中将该技术应用于战术作战, 达到了在大量策略上可靠且低方差的效果。[Andrade et al, 2004] 在一个简单的格斗游戏 “Knock'em” 中对 Q 学习的学习器应用了类似的技术: 如果学习器领先太多, 那么几个最好的动作 (按 Q 值排列) 就被禁用了。由于使用了 Q 函数, 难度适配可以在线修改。玩家的生命值之间的区别决定了领导者以及游戏是否太难了。[Hunicke and Chapman, 2004] 提出了一种修改游戏环境的难度调整系统, 例如, 当游戏太难时, 向玩家提供更多的武器并产生较少的敌人。第一人称射击游戏 “Max Payne”^⑤ 中实现了类似的系统, 能根据玩家的表现对敌人的健康状况和自动瞄准辅助的数量进行调整。[Hagelbäck and Johansson, 2009] 在 RTS 游戏中针对的是具体得分差异。根据调查问卷, 与静态对手相比, 他们的方法显著提高了乐趣和感知到的变化。

17.4.1.2 游戏过程中的适应

569 强化学习可以用于玩家风格的在线适应。原则上, 这可以防止玩家一遍又一遍地应用相同的技巧, 迫使其产生更多样化的玩法, 并提供更具挑战性的对手。它还可以防止硬编码的对手策略的易探索性和脆弱性。[Spronck et al, 2006] 通过在 RPG 游戏 “无冬之夜” (Neverwinter night) 中使用用于战术作战的动态脚本来展示这一想法。通过动态脚本训练的 AI 在 20 ~ 30 轮之后学会战胜固定的 AI。

除了 Black & White 和 Creatures 以及学术游戏 NERO [Stanley et al, 2005] 外, 极少有视频游戏具备游戏过程中的适应性 (in-game adaptation)。这三个游戏有许多共同点: 学习是游戏的核心部分, 玩家可以观察他的角色与环境的交互, 还可以通过奖励 / 惩罚并修改环境来训练他的角色 (例如, 设置任务或提供新的刺激)。

极少有在游戏过程中学习的例子的原因主要有两个: 学习速度和可靠性。要有任何明显的效果, 学习应该在几次重复 (或最多几十次) 之后可见。[Spronck et al, 2006] 的动态脚本应用在这方面处于可用性的边缘。对于可靠性, 学习受到包括随机性在内的许多因素的影响。因此, 学习过程如何结束是不可预测的, 结果可能会有明显的差异和失败的可能。可能除了如 Black & White 这样玩家控制学习过程的 “教学游戏” 之外, 这种不可预测性在商业游戏中都是不被接受的。

17.4.2 开发期间的学习

如果在游戏发布之前完成所有学习然后由开发人员测试和修正所产生的策略, 那么就保留了大量离线适应的机会。例如, 在赛车游戏 Re-Volt、Colin McRae Rally 和 Forza Motorsport 中, 对手都由 AI 技术训练。我们强调两个离线学习能有可观结果的领域: 微观管理和游戏平衡。

17.4.2.1 微观管理和辅导员

即时战略游戏和回合制策略游戏 (如文明系列) 的一个主要问题是需要进行微观管理 [Wender and Watson, 2009]: 在进行高层决策 (攻击哪里、研究什么技术) 的同时, 玩家需要做出较低层次的决策, 如为每个城市分配生产和调拨劳动力并管理单个的士兵和工人。微

⑤ http://www.gameontology.org/index.php/Dynamic_Difficulty_Adjustment.

观管理任务是乏味且重复的。在回合制策略游戏中，避免微观管理的一个标准解决方案是租用可以参与工作的 AI 辅导员；在 RTS 游戏中，辅导员还不常见。[Szczepański and Aamodt, 2009] 在 RTS 游戏魔兽争霸 3 中将基于案例的学习应用于微观管理。根据他们的实验，学习器（例如，学会了在战斗中治疗伤兵）受到初级和中级玩家的好评（虽然不是来自那些善于微观管理的专家，而且被重写的命令所迷惑）。[Sharifi et al, 2010] 使用 SARSA (λ) 在 RPG 游戏无冬之夜中训练伴侣动作。伴侣从两个来源获得奖励：对成功 / 失败完成任务的“内部”奖励（例如对拆除陷阱尝试失败的惩罚），以及来自玩家的可选的、稀少的“口头奖励” (verbal reward)。

17.4.2.2 游戏平衡

游戏平衡 (game balance) 可以理解为两件事情：

- 游戏中采用不对称的设计让对手间获胜的机会比较平衡。不对称可以是轻微的，例如，在一个 RTS 游戏中，不同的种族以不同的优势和劣势开始。或者更强壮，就像在桌游 “Last Night on Earth” 中，一个玩家控制弱小但数量充足的僵尸，而另一个控制着一小群（相当强大）的幸存者。
- 单个玩家的策略之间的平衡。

第一种平衡对于公平显然是必要的，而第二种对于确保玩家每次都不会被迫重复一个主导策略是非常重要的，但是游戏中确实有更多种类的选择，以便玩家做出有意义的决定，同时可能有更多的乐趣。两种平衡方式都不容易建立，需要进行广泛的测试，而且在测试过程中可能仍没有发现失衡。这的确发生过，例如，在 RTS 游戏 Command and Conquer 中的坦克冲锋，或者是可收集的纸牌游戏 Magic: The Gathering，其中一些牌太过强大以至于不得不在比赛中限制或禁用它们。[Kalles and Kanellopoulos, 2001] 使用 TD 学习来测试抽象棋盘游戏的平衡度。

17.5 总结

游戏是强化学习研究的一个有大量应用的蓬勃发展着的领域。时差学习、蒙特卡罗树搜索以及演化强化学习都是最受欢迎的技术之一。在许多案例中，强化学习方法与其他 AI 技术和人工智能竞争，而在其他方面，虽尚未发生但改进很快。

本章试图给出游戏特有的问题和挑战的代表性案例，以及游戏应用中使用的主要方法。在评论中，我们专注于理解造成有效与无效的强化学习算法之间区别的技巧和诀窍。我们决不会声称完全介绍了游戏中使用的强化学习技术。我们甚至没有试图给出历史沿革的概述，而且我们不得不排除与演化计算、博弈论、监督学习和一般 AI 研究相关的大多数成果。此外，由于本章的重点是强化学习与游戏的关系这一侧面，我们不得不省略完整的游戏类型，如回合制策略游戏、角色扮演游戏，或（从更理论的角度）迭代矩阵游戏。幸运的是，有很多关于这些话题的高质量的综述文章。对于传统的棋盘游戏和卡牌游戏，[Fürrnkranz, 2001; Fürrnkranz, 2007] 着重于机器学习技术，[Schaeffer, 2000]；van den Herik et al, 2002；Mańdziuk, 2010] 对包括机器学习（以及强化学习）在内的通用人工智能方法进行了评述。[Ghory, 2004] 很好地概述了适用于传统棋盘游戏的 TD 变体，而 [Galway et al, 2008] 着眼于演化方法和电脑游戏。

可能我们学习到的最重要的教训是：强化学习技术是强大的，但为了使其工作，支持组件同样重要，包括一个适当的表示、训练计划、特定游戏的领域知识和启发信息。虽然这

些东西本身不足以创建强大的游戏学习器，普通形式的基本强化学习方法也不够——即使是 TD-Gammon 也需要其他组件的支持 [Tesauro, 2002]。

开始研究强化学习和游戏的门槛较低。许多游戏以及一些支撑的 AI 组件都可以轻松得到。大多数传统棋盘和卡牌游戏都有强大的开源引擎，许多电脑游戏也存在开源副本。像 Atari 2600 这样的旧版游戏系统的模拟器可以接入数百个老一代的视频游戏。此外，许多最先进的游戏都可以通过 API 或脚本语言访问它们的 AI 系统，允许相对容易地实现学习器。最后，多项比赛提供了将 AI 实现与电脑游戏进行比较的场所，例如，每年在 IEEE CIG 会议上组织的 Pac-Man 比赛^①、马里奥 AI 锦标赛^②，或者早期强化学习比赛中的俄罗斯方块、马里奥和 RTS 领域的游戏^③。

强化学习是一个有前景的研究领域，有很多挑战及开放的问题需要解决。游戏是令人兴奋的，其中许多还没有被 AI 攻克。每个人的机会都很多。

致谢。作者衷心地感谢 Csaba Szepesvári、Mike Bowling、Thomas Degris、Gábor Balázs、Joseph Modayil 和 Hengshuai Yao 的帮助。他们的意见和建议极大地改进了这一章，就像匿名评论者的评论一样有用。

参考文献

- Aha, D.W., Molineaux, M., Ponsen, M.: Learning to win: Case-based plan selection in a real-time strategy game. *Case-Based Reasoning Research and Development*, 5–20 (2005)
- Amit, A., Markovitch, S.: Learning to bid in bridge. *Machine Learning* 63(3), 287–327 (2006)
- Andrade, G., Santana, H., Furtado, A., Leitão, A., Ramalho, G.: Online adaptation of computer games agents: A reinforcement learning approach. *Scientia* 15(2) (2004)
- Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. *Machine Learning* 47, 235–256 (2002)
- Bartók, G., Szepesvári, C., Zilles, S.: Models of active learning in group-structured state spaces. *Information and Computation* 208, 364–384 (2010)
- Baxter, J., Tridgell, A., Weaver, L.: Learning to play chess using temporal-differences. *Machine learning* 40(3), 243–263 (2000)
- Baxter, J., Tridgell, A., Weaver, L.: Reinforcement learning and chess. In: *Machines that learn to play games*, pp. 91–116. Nova Science Publishers, Inc. (2001)
- Beal, D., Smith, M.C.: Learning piece values using temporal differences. *ICCA Journal* 20(3), 147–151 (1997)
- Bertsekas, D.P., Tsitsiklis, J.N.: *Neuro-Dynamic Programming*. Athena Scientific (1996)
- Billings, D., Davidson, A., Schauenberg, T., Burch, N., Bowling, M., Holte, R.C., Schaeffer, J., Szafron, D.: Game-Tree Search with Adaptation in Stochastic Imperfect-Information Games. In: van den Herik, H.J., Björnsson, Y., Netanyahu, N.S. (eds.) *CG 2004. LNCS*, vol. 3846, pp. 21–34. Springer, Heidelberg (2006)
- Björnsson, Y., Finnsson, H.: Cadiaplayer: A simulation-based general game player. *IEEE Transactions on Computational Intelligence and AI in Games* 1(1), 4–15 (2009)
- Böhm, N., Kókai, G., Mandl, S.: Evolving a heuristic function for the game of tetris. In: *Proc. Lernen, Wissensentdeckung und Adaptivität LWA*, pp. 118–122 (2004)
- Boumaza, A.: On the evolution of artificial Tetris players. In: *IEEE Symposium on Computational Intelligence and Games* (2009)
- Bouzy, B., Helmstetter, B.: Monte Carlo Go developments. In: *Advances in Computer Games*, pp. 159–174 (2003)

① <http://cswwww.essex.ac.uk/staff/sml/pacman/PacManContest.html>

② <http://www.marioai.org/>

③ <http://rl-competition.org/>

- Bowling, M.: Convergence and no-regret in multiagent learning. In: *Neural Information Processing Systems*, pp. 209–216 (2004)
- Buro, M.: From simple features to sophisticated evaluation functions. In: *International Conference on Computers and Games*, pp. 126–145 (1998)
- Buro, M., Furtak, T.: RTS games as test-bed for real-time research. *JCIS*, 481–484 (2003)
- Buro, M., Lanctot, M., Orsten, S.: The second annual real-time strategy game AI competition. In: *GAME-ON NA* (2007)
- Chaslot, G., Winands, M., Herik, H., Uiterwijk, J., Bouzy, B.: Progressive strategies for monte-carlo tree search. *New Mathematics and Natural Computation* 4(3), 343 (2008)
- Chaslot, G., Fiter, C., Hoock, J.B., Rimmel, A., Teytaud, O.: Adding Expert Knowledge and Exploration in Monte-Carlo Tree Search. In: van den Herik, H.J., Spronck, P. (eds.) *ACG 2009*. LNCS, vol. 6048, pp. 1–13. Springer, Heidelberg (2010)
- Chatriot, L., Gelly, S., Jean-Baptiste, H., Perez, J., Rimmel, A., Teytaud, O.: Including expert knowledge in bandit-based Monte-Carlo planning, with application to computer-Go. In: *European Workshop on Reinforcement Learning* (2008)
- Coquelin, P.A., Munos, R.: Bandit algorithms for tree search. In: *Uncertainty in Artificial Intelligence* (2007)
- Coulom, R.: Efficient Selectivity and Backup Operators in Monte-carlo Tree Search. In: van den Herik, H.J., Ciancarini, P., Donkers, H.H.L.M.(J.) (eds.) *CG 2006*. LNCS, vol. 4630, pp. 72–83. Springer, Heidelberg (2007)
- Coulom, R.: Computing Elo ratings of move patterns in the game of go. *ICGA Journal* 30(4), 198–208 (2007)
- Dahl, F.A.: Honte, a Go-playing program using neural nets. In: *Machines that learn to play games*, pp. 205–223. Nova Science Publishers (2001)
- Davidson, A.: Opponent modeling in poker: Learning and acting in a hostile and uncertain environment. Master's thesis, University of Alberta (2002)
- Diuk, C., Cohen, A., Littman, M.L.: An object-oriented representation for efficient reinforcement learning. In: *International Conference on Machine Learning*, pp. 240–247 (2008)
- Droste, S., Fürnkranz, J.: Learning of piece values for chess variants. Tech. Rep. TUD-KE–2008-07, Knowledge Engineering Group, TU Darmstadt (2008)
- Džeroski, S., Raedt, L.D., Driessens, K.: Relational reinforcement learning. *Machine Learning* 43(1-2), 7–52 (2001)
- Epstein, S.L.: Toward an ideal trainer. *Machine Learning* 15, 251–277 (1994)
- Farias, V.F., van Roy, B.: Tetris: A Study of Randomized Constraint Sampling. In: *Probabilistic and Randomized Methods for Design Under Uncertainty*. Springer, UK (2006)
- Fawcett, T., Utgoff, P.: Automatic feature generation for problem solving systems. In: *International Conference on Machine Learning*, pp. 144–153 (1992)
- Finkelstein, L., Markovitch, S.: Learning to play chess selectively by acquiring move patterns. *ICCA Journal* 21, 100–119 (1998)
- Fudenberg, D., Levine, D.K.: *The theory of learning in games*. MIT Press (1998)
- Fürnkranz, J.: Machine learning in games: a survey. In: *Machines that Learn to Play Games*, pp. 11–59. Nova Science Publishers (2001)
- Fürnkranz, J.: Recent advances in machine learning and game playing. Tech. rep., TU Darmstadt (2007)
- Galway, L., Charles, D., Black, M.: Machine learning in digital games: a survey. *Artificial Intelligence Review* 29(2), 123–161 (2008)
- Gelly, S., Silver, D.: Achieving master-level play in 9x9 computer go. In: *AAAI*, pp. 1537–1540 (2008)
- Gelly, S., Wang, Y., Munos, R., Teytaud, O.: Modification of UCT with patterns in Monte-Carlo go. Tech. rep., INRIA (2006)
- Gherry, M.: A game-learning machine. PhD thesis, University of California, San Diego, CA (1993)
- Ghory, I.: Reinforcement learning in board games. Tech. rep., Department of Computer Science, University of Bristol (2004)
- Gilgenbach, M.: Fun game AI design for beginners. In: *AI Game Programming Wisdom*, vol. 3. Charles River Media, Inc. (2006)

- Gilpin, A., Sandholm, T.: Lossless abstraction of imperfect information games. *Journal of the ACM* 54(5), 25 (2007)
- Gilpin, A., Sandholm, T., Sørensen, T.B.: Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold'em poker. In: *AAAI*, vol. 22, pp. 50–57 (2007)
- Ginsberg, M.L.: *Gib: Imperfect information in a computationally challenging game*. *Journal of Artificial Intelligence Research* 14, 313–368 (2002)
- Gould, J., Levinson, R.: Experience-based adaptive search. Tech. Rep. UCSC-CRL-92-10, University of California at Santa Cruz (1992)
- Günther, M.: Automatic feature construction for general game playing. PhD thesis, Dresden University of Technology (2008)
- Hagelbäck, J., Johansson, S.J.: Measuring player experience on runtime dynamic difficulty scaling in an RTS game. In: *International Conference on Computational Intelligence and Games* (2009)
- Hartley, T., Mehdi, Q., Gough, N.: Online learning from observation for interactive computer games. In: *International Conference on Computer Games: Artificial Intelligence and Mobile Systems*, pp. 27–30 (2005)
- van den Herik, H.J., Uiterwijk, J.W.H.M., van Rijswijck, J.: Games solved: Now and in the future. *Artificial Intelligence* 134, 277–311 (2002)
- Hsu, F.H.: *Behind Deep Blue: Building the Computer that Defeated the World Chess Champion*. Princeton University Press, Princeton (2002)
- Hunnicke, R., Chapman, V.: AI for dynamic difficult adjustment in games. In: *Challenges in Game AI Workshop* (2004)
- Kakade, S.: A natural policy gradient. In: *Advances in Neural Information Processing Systems*, vol. 14, pp. 1531–1538 (2001)
- Kalles, D., Kanellopoulos, P.: On verifying game designs and playing strategies using reinforcement learning. In: *ACM Symposium on Applied Computing*, pp. 6–11 (2001)
- Kerbusch, P.: Learning unit values in Wargus using temporal differences. BSc thesis (2005)
- Kocsis, L., Szepesvári, C.: Bandit Based Monte-Carlo Planning. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) *ECML 2006. LNCS (LNAI)*, vol. 4212, pp. 282–293. Springer, Heidelberg (2006)
- Kocsis, L., Szepesvári, C., Winands, M.H.M.: RSPSA: Enhanced Parameter Optimization in Games. In: van den Herik, H.J., Hsu, S.-C., Hsu, T.-s., Donkers, H.H.L.M.(J.) (eds.) *CG 2005. LNCS*, vol. 4250, pp. 39–56. Springer, Heidelberg (2006)
- Kok, E.: Adaptive reinforcement learning agents in RTS games. Master's thesis, University of Utrecht, The Netherlands (2008)
- Koza, J.: *Genetic programming: on the programming of computers by means of natural selection*. MIT Press (1992)
- Kuhlmann, G.J.: Automated domain analysis and transfer learning in general game playing. PhD thesis, University of Texas at Austin (2010)
- Lagoudakis, M.G., Parr, R., Littman, M.L.: Least-Squares Methods in Reinforcement Learning for Control. In: Vlahavas, I.P., Spyropoulos, C.D. (eds.) *SETN 2002. LNCS (LNAI)*, vol. 2308, pp. 249–260. Springer, Heidelberg (2002)
- Laursen, R., Nielsen, D.: Investigating small scale combat situations in real time strategy computer games. Master's thesis, University of Aarhus (2005)
- Levinson, R., Weber, R.: Chess Neighborhoods, Function Combination, and Reinforcement Learning. In: Marsland, T., Frank, I. (eds.) *CG 2001. LNCS*, vol. 2063, pp. 133–150. Springer, Heidelberg (2002)
- Lorenz, U.: Beyond Optimal Play in Two-Person-Zerogame Games. In: Albers, S., Radzik, T. (eds.) *ESA 2004. LNCS*, vol. 3221, pp. 749–759. Springer, Heidelberg (2004)
- Mańdziuk, J.: *Knowledge-Free and Learning-Based Methods in Intelligent Game Playing*. Springer, Heidelberg (2010)
- Marthi, B., Russell, S., Latham, D.: Writing Strategus-playing agents in concurrent alisp. In: *IJCAI Workshop on Reasoning, Representation, and Learning in Computer Games*, pp. 67–71 (2005)
- McGlinchey, S.J.: Learning of AI players from game observation data. In: *GAME-ON*, pp.

- 106–110 (2003)
- Molineaux, M., Aha, D.W., Ponsen, M.: Defeating novel opponents in a real-time strategy game. In: IJCAI Workshop on Reasoning, Representation, and Learning in Computer Games, pp. 72–77 (2005)
- Moriarty, D.E., Miikkulainen, R.: Discovering complex Othello strategies through evolutionary neural networks. *Connection Science* 7, 195–209 (1995)
- Müller, M.: Position evaluation in computer go. *ICGA Journal* 25(4), 219–228 (2002)
- Naddaf, Y.: Game-independent AI agents for playing Atari 2600 console games. Master's thesis, University of Alberta (2010)
- Pollack, J.B., Blair, A.D.: Why did TD-Gammon work? In: *Neural Information Processing Systems*, vol. 9, pp. 10–16 (1997)
- Ponsen, M., Spronck, P.: Improving adaptive game AI with evolutionary learning. In: *Computer Games: Artificial Intelligence, Design and Education* (2004)
- Ponsen, M., Muñoz-Avila, H., Spronck, P., Aha, D.W.: Automatically acquiring adaptive real-time strategy game opponents using evolutionary learning. In: *Proceedings of the 17th Innovative Applications of Artificial Intelligence Conference* (2005)
- Ponsen, M., Spronck, P., Tuyls, K.: Hierarchical reinforcement learning in computer games. In: *Adaptive Learning Agents and Multi-Agent Systems*, pp. 49–60 (2006)
- Ponsen, M., Taylor, M.E., Tuyls, K.: Abstraction and Generalization in Reinforcement Learning: A Summary and Framework. In: Taylor, M.E., Tuyls, K. (eds.) *ALA 2009. LNCS*, vol. 5924, pp. 1–33. Springer, Heidelberg (2010)
- Ramanujan, R., Sabharwal, A., Selman, B.: Adversarial search spaces and sampling-based planning. In: *International Conference on Automated Planning and Scheduling* (2010)
- Risk, N., Szafron, D.: Using counterfactual regret minimization to create competitive multi-player poker agents. In: *International Conference on Autonomous Agents and Multiagent Systems*, pp. 159–166 (2010)
- Rubin, J., Watson, I.: Computer poker: A review. *Artificial Intelligence* 175(5–6), 958–987 (2011)
- Schaeffer, J.: The games computers (and people) play. In: Zelkowitz, M. (ed.) *Advances in Computers*, vol. 50, pp. 89–266. Academic Press (2000)
- Schaeffer, J., Hlynka, M., Jussila, V.: Temporal difference learning applied to a high-performance game-playing program. In: *International Joint Conference on Artificial Intelligence*, pp. 529–534 (2001)
- Schnizlein, D., Bowling, M., Szafron, D.: Probabilistic state translation in extensive games with large action sets. In: *International Joint Conference on Artificial Intelligence*, pp. 278–284 (2009)
- Schraudolph, N.N., Dayan, P., Sejnowski, T.J.: Learning to evaluate go positions via temporal difference methods. In: *Computational Intelligence in Games. Studies in Fuzziness and Soft Computing*, ch. 4, vol. 62, pp. 77–98. Springer, Heidelberg (2001)
- Scott, B.: The illusion of intelligence. In: *AI Game Programming Wisdom*, pp. 16–20. Charles River Media (2002)
- Shapiro, A., Fuchs, G., Levinson, R.: Learning a Game Strategy Using Pattern-Weights and Self-Play. In: Schaeffer, J., Müller, M., Björnsson, Y. (eds.) *CG 2002. LNCS*, vol. 2883, pp. 42–60. Springer, Heidelberg (2003)
- Sharifi, A.A., Zhao, R., Szafron, D.: Learning companion behaviors using reinforcement learning in games. In: *AIIDE* (2010)
- Sharma, S., Kobti, Z., Goodwin, S.: General game playing: An overview and open problems. In: *International Conference on Computing, Engineering and Information*, pp. 257–260 (2009)
- Silver, D., Tesauro, G.: Monte-carlo simulation balancing. In: *International Conference on Machine Learning* (2009)
- Silver, D., Sutton, R., Mueller, M.: Sample-based learning and search with permanent and transient memories. In: *ICML* (2008)
- Spronck, P., Sprinkhuizen-Kuyper, I., Postma, E.: Difficulty scaling of game AI. In: *GAME-ON 2004: 5th International Conference on Intelligent Games and Simulation* (2004)
- Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I., Postma, E.: Adaptive game AI with dynamic scripting. *Machine Learning* 63(3), 217–248 (2006)

- Stanley, K.O., Bryant, B.D., Miikkulainen, R.: Real-time neuroevolution in the NERO video game. *IEEE Transactions on Evolutionary Computation* 9(6), 653–668 (2005)
- Sturtevant, N., White, A.: Feature construction for reinforcement learning in Hearts. In: *Advances in Computers and Games*, pp. 122–134 (2007)
- Szczepański, T., Aamodt, A.: Case-based reasoning for improved micromanagement in real-time strategy games. In: *Workshop on Case-Based Reasoning for Computer Games*, 8th International Conference on Case-Based Reasoning, pp. 139–148 (2009)
- Szita, I., Lőrincz, A.: Learning Tetris using the noisy cross-entropy method. *Neural Computation* 18(12), 2936–2941 (2006a)
- Szita, I., Lőrincz, A.: Learning to play using low-complexity rule-based policies: Illustrations through Ms. Pac-Man. *Journal of Artificial Intelligence Research* 30, 659–684 (2006b)
- Szita, I., Szepesvári, C.: Sz-tetris as a benchmark for studying key problems of rl. In: *ICML 2010 Workshop on Machine Learning and Games* (2010)
- Szita, I., Chaslot, G., Spronck, P.: Monte-Carlo Tree Search in Settlers of Catan. In: van den Herik, H.J., Spronck, P. (eds.) *ACG 2009. LNCS*, vol. 6048, pp. 21–32. Springer, Heidelberg (2010)
- Tesauro, G.: Practical issues in temporal difference learning. *Machine Learning* 8, 257–277 (1992)
- Tesauro, G.: Temporal difference learning and TD-gammon. *Communications of the ACM* 38(3), 58–68 (1995)
- Tesauro, G.: Comments on co-evolution in the successful learning of backgammon strategy'. *Machine Learning* 32(3), 241–243 (1998)
- Tesauro, G.: Programming backgammon using self-teaching neural nets. *Artificial Intelligence* 134(1-2), 181–199 (2002)
- Thiery, C., Scherrer, B.: Building controllers for Tetris. *ICGA Journal* 32(1), 3–11 (2009)
- Thrun, S.: Learning to play the game of chess. In: *Neural Information Processing Systems*, vol. 7, pp. 1069–1076 (1995)
- Utgoff, P.: Feature construction for game playing. In: Fürnkranz, J., Kubat, M. (eds.) *Machines that Learn to Play Games*, pp. 131–152. Nova Science Publishers (2001)
- Utgoff, P., Precup, D.: Constructive function approximation. In: Liu, H., Motoda, H. (eds.) *Feature Extraction, Construction and Selection: A Data Mining Perspective*, vol. 453, pp. 219–235. Kluwer Academic Publishers (1998)
- Veness, J., Silver, D., Uther, W., Blair, A.: Bootstrapping from game tree search. In: *Neural Information Processing Systems*, vol. 22, pp. 1937–1945 (2009)
- Weber, B.G., Mateas, M.: Case-based reasoning for build order in real-time strategy games. In: *Artificial Intelligence and Interactive Digital Entertainment*, pp. 1313–1318 (2009)
- Wender, S., Watson, I.: Using reinforcement learning for city site selection in the turn-based strategy game Civilization IV. In: *Computational Intelligence and Games*, pp. 372–377 (2009)
- Wiering, M.A.: Self-play and using an expert to learn to play backgammon with temporal difference learning. *Journal of Intelligent Learning Systems and Applications* 2, 57–68 (2010)
- Zinkevich, M., Johanson, M., Bowling, M., Piccione, C.: Regret minimization in games with incomplete information. In: *Neural Information Processing Systems*, pp. 1729–1736 (2008)

机器人领域的强化学习综述

Jens Kober, Jan Peters

摘要

由于大多数的自动化机器人的动作生成问题可以概括为序列决策问题，因此，机器人领域为强化学习提供了大量重要并且有趣的应用平台。无独有偶，机器人领域中的真实世界挑战给强化学习的验证提供了大量的应用场景。因此，这两个学科之间的相互作用可以视为如同物理与数学之间的作用那么有前途。然而，只有小部分强化学习领域内的科学家能够协助机器人解决在监督过程中遇到的大多数问题。因此，我们将会介绍机器人强化学习领域内需要面对的大多数重要的挑战，以引起大家重视。为了达成这个目标，我们将会调研大量已经成功将强化学习应用到真实机器人的动作生成的工作。尽管这个领域非常复杂，我们仍会讨论如何呈现已经成功并且易于驾驭的方法以及将会学习如何表达或者先验知识是如何对结果产生重要作用的。因此，本章的重点是聚焦在基于模型和无模型之间的选择，同样也是基于价值函数以及策略搜索方法之间的选择。因此，我们获得了一个相当完整的关于机器人强化学习的综述，这些结果应该有助于大多数强化学习领域内的研究者更好地理解这一领域。

18.1 简介

机器人领域有着近乎无穷个有趣的学习问题，其中一大部分可以概括为强化学习问题。[579]图 18-1 说明了各式各样的机器人使用强化学习来学习任务。然而，机器人领域与良好定义的经典强化学习基准问题有很大区别，典型强化学习基准问题通常具有离散的状态和动作。与此相反，机器人领域中的许多现实问题通常最好用高维度和连续状态以及动作表达。因为每一个单独的实验都是昂贵的，因此，这类应用迫使我们关注那些不是经常出现在经典的强化学习基准例子中的问题。在本章中，我们将重点介绍机器人强化学习面临的挑战以及这些挑战给这一领域带来的问题。

机器人的高维度特点是由于许多现代拟人机器人的自由度决定的。真实系统是十分昂贵的，并且其经验通常难以重现。然而，真实系统是不能由仿真来取代的，至少对于高度动态的任务来说，因为小的模型错误的积累导致显著不同的动态动作。另一个机器人强化学习领域面临的挑战就是生成适当的奖励函数。优良的奖励将会引导系统迅速而成功地处理真实场景的经验，但是这需要大量的人工付出。

显然并不是所有的强化学习方法都同样适用于机器人领域。事实上，扩展到更有趣的任务的许多方法是基于模型的 [Atkeson et al, 1997; Abbeel et al, 2007] 而且通常是使用策略搜索而不是基于价值函数的方法 [Gullapalli et al, 1994; Miyamoto et al, 1996; Kohl and Stone, 2004; Tedrake et al, 2005; Peters and Schaal, 2008a, b; Kober and Peter, 2009]。这与主流的强化学习形成了鲜明对比 [Kaelbling et al, 1996; Sutton and Barto, 1998]。我们主要会在方法

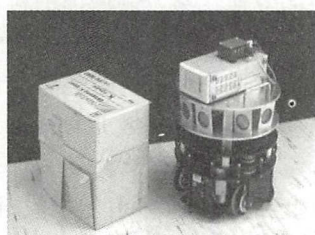
论层面上尝试通过引用最原始的论文来给真实机器人的强化学习一个相对完整的概括。

由于目前没有提出能够较简单地延伸至机器人领域的方法，我们将会介绍如何使机器人强化学习方法变为可行。对于这个问题，我们提出了几种方法，比如说为你的价值函数或者策略选择一个合适的表达方式，结合相应的先验知识以及从模拟仿真中迁移学习。

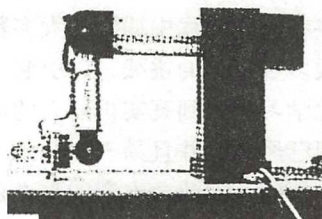
在这一章，我们将会综述真实机器人强化学习并且突出这些方法是如何能够处理相应的挑战。我们将不太关注那些由于略有增强的网络世界或者仅仅从仿真模拟中学习而来的结果。应用强化学习技术到机器人领域内的挑战将会在 18.2 节中讨论。

标准的强化学习方法已经受到了所提及的各种挑战。正如已经指出这一点的强化学习论文 [Kaelbling et al, 1996] 所说，“我们必须放弃白板学习技术并且开始考虑能够给学习过程带来影响的偏差”。在 18.3 节将会简明地介绍介绍机器人领域内的强化学习技术，而在 18.4 节到 18.6 节将会介绍不同的能够使强化学习方法变为可行的方法。在 18.7 节中，我们将会提供一个球入杯的样例，来突出本章讨论的多种方法中哪些会在解决该复杂问题时特别有用。在 18.8 节中，我们将会给出结论并展望相关有趣的问题。

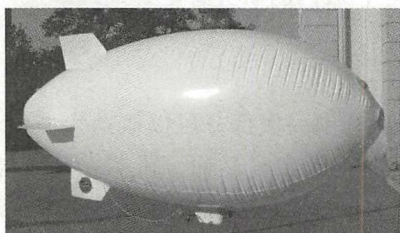
[580]



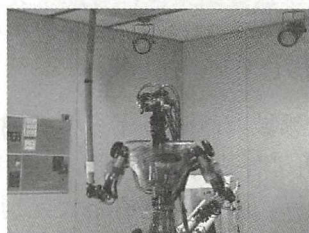
a) OBELIX 机器人



b) 斑马零式机器人



c) 自动驾驶的飞艇



d) 萨科斯人形机器人

图 18.1 这张图片展示了已经应用了强化学习的机器人。这些机器人覆盖了从轮式移动机器人、机器人手臂、自动化车辆到人形机器人的所有范围。a) OBELIX 机器人（图片转载经过 SridharMahadevan 许可）是一个通过基于价值函数的方法 [Mahadevan and Connell, 1992] 学会推动箱子的轮式移动机器人。b) 斑马零式机器人（图片转载经过 Rod Grupen 许可）通过无模型策略梯度方法 [Gullapalli et al, 1994] 学会了插入轴孔的任务。c) 这样的自动驾驶飞艇（图片转载经过 Axel Rottmann 许可）的控制技术是通过基于价值函数 [Rottmann et al, 2007] 和基于模型的策略搜索 [Ko et al, 2007] 实现的。d) 萨科斯人形机器人（图片转载经过 Stefan Schaal 许可）通过使用前向模型 (Schaal, 1997) 的方法学会了平衡杆的任务

18.2 机器人强化学习中的挑战

强化学习一般来说是一个困难的问题，它的许多难点也恰恰在机器人的领域内体现出

来。大多数机器人的状态和动作在本质上是连续和高维度的，这也就是我们将会面对的“维度灾难” [Bellman, 1957]，详见 18.2.1 节。当我们处理复杂的物理系统时，由于任务的执行时间过长，并且需要人为的介入，比如维护保养等（详见 18.2.2 节），这就造成了样例成本非常昂贵。一个机器人要求的算法必须能实时运行，并且能够处理物理系统固有的感知延迟和执行延迟（详见 18.2.3 节）。显而易见，仿真模拟可以减少许多问题的出现，但是由于存在模型误差，如何从仿真模拟迁移到真实的机器人是个挑战，详见 18.2.4 节。还有一个经常被低估的问题就是目标规范，这个问题可以通过设计一个好的奖励函数来解决。正如 18.2.5 节中所强调的，选择的不同可能造成可行与不合理探索截然迥异的差异。

581

18.2.1 维度灾难

当 [Bellman, 1957] 在探究连续状态空间中的最佳控制时，其面临一个指数爆炸级别的离散状态并因此发明了维度灾难这个词。机器人系统本质上是用来处理连续的状态和动作的，比如说，如图 18.2 所示，在拍球任务中，一个恰当的机器人状态的表示应该包括它的关节角度和七个自由度的速度以及球的笛卡尔坐标及速度。机器人的动作产生通常来自于电机的指令，例如力矩或者加速度。在这个例子中，我们有 $2 \times (7+3) = 20$ 个状态维度和 7 维的连续动作。显然，其他任务可能需要更多的维度，例如，类似于人类的动作常常遵从对立的原则，另外还允许控制刚度。显然这样的维度对于机器人和强化学习社区来说都是巨大挑战。

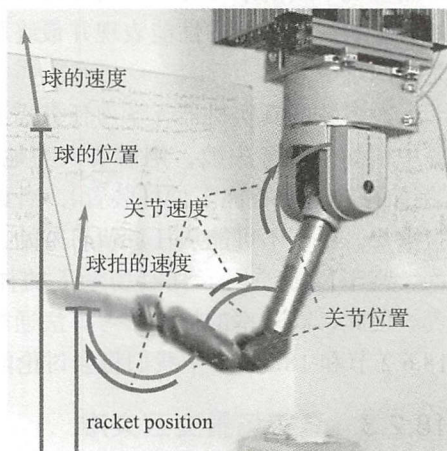


图 18.2 这张图片展示了机器人强化学习任务中的状态空间

在机器人中，像这类问题的任务通常是通过机器人工程师降低机器人的复杂性，仅仅实现一些较低层次的功能来完成的。在拍球的样例中，我们可以通过操作空间控制法则来控制机器人在球拍的运动空间内（这是一个低维度的空间，因为球拍的运动方向是沿着绳子的定点线性不变的）运动，以达到简化的效果 [Nakanishi et al, 2008]。许多商业型机器人系统都会封装一些状态和动作组件到嵌入式控制系统中（例如，轨迹片段通常都是作为工业机器人的动作）；然而，根据我们的经验 [Schaal et al, 2002 ; Peter et al, 2010b]，这种降低维度状态的方法严重限制了机器人的动态能力。

582

强化学习社区在使用计算抽象处理维度上有着悠久的历史。它提供了更大的适用工具集，这些工具包括了自适应离散化 [Busoniu et al, 2010]、函数近似方法 [Sutton and Barto, 1998] 以及宏操作或者宏选项 [Barto and Mahadevan, 2003]。宏操作将会允许任务分解给基本部件并使其易于移植到机器人中。比如，一个宏操作“向左移动一米”的完成可以通过一个更低级的控制器来实现，这个控制器需要在保证精度的同时还要注意到加速、移动、停止等操作。对于移动机器人来说，标准的强化学习方法使用一个有限的手动生成的宏操作可以更好地完成导航任务。然而，对于这种方法来说，关键问题就是如何自动生成相应的宏操作。在 18.4 节中，我们将会讨论这些已经成功应用到机器人强化学习中的方法。

18.2.2 真实场景样本灾难

机器人的本质就是要与真实场景互动,因此,机器人强化学习能够从大部分现实问题中学习。例如,机器人硬件因为存在自然磨损消耗,并且需要非常小心的维护,因此是十分昂贵的。修理一个机器人系统必须考虑到成本、体力劳动和长时间的等待等不可忽视的成本。因此,为了将强化学习应用到机器人中,如何安全地探索(safe exploration)就成了学习过程的关键问题,这也是一般强化学习社区都忽略的问题。

然而,几个有关真实场景的问题使得机器人成为了富有挑战的领域。由于一个机器人的动态性会随许多外部因素而改变,例如温度以及物理磨损,这整个学习过程可能永远不会完全收敛。例如,它可能需要跟踪方案(tracking solution)[Sutton et al, 2007]。一般来说,不能重现早期的学习环节中的环境变量,并且外部因素的影响也难以知晓,例如光照因素是如何影响视觉系统的性能表现并最终影响到整个任务的结果。这个问题使得算法的比较变得尤其困难。

大多数的真实机器人学习任务需要人为的监管,例如,在图 18-1d 中的机器人平衡杆任务中,如果任务失败,则需要人为地将杆放回机器人的末端。即使有自动复位按钮的存在(例如存在一个智能复位杆装置),当真实机器人无法加快任务的完成时,学习的速度变得尤为重要。由于不可能从任意状态开始,就要求整个过程比较完备。

鉴于以上原因,真实场景下的样本会由于时间、劳动或者是财务等原因变得非常昂贵。因此,高效率的样本学习算法通过保证使机器人从少量的实验中学学习变得十分重要。在 18.6.2 节和 18.6.3 节中我们将会讨论能够减少与真实场景之间的交互的几种方法。

18.2.3 真实场景交互灾难

机器人是一个在学习算法与机器初始化设置的交互上有着严格约束的物理系统。通常来讲,机器人需要保持固定频率接受指令,并且对于动态任务来说动作不能被打断。因此,学习器需要实时选择下一步动作。通常来讲是不可能停下来去考虑下一步,每一步动作之间的学习或者计划都是需要通过学习算法在一个固定时间内做出处理。因此,如果算法的运行过程依赖于样本的数量,不仅仅实验样本的获取是十分昂贵的,而且可用的样本数量也是十分有限的。这些限制在情景设置过程中就要缓和很多,其中对时间密集部分的学习可以推迟到情景之间的时间段。

物理系统中的感知与执行通常都会存在延迟。由于处理延迟以及通信延迟,由传感器显示的状态通常会轻微落后于真实的状态。更关键的是,执行过程中存在通信延迟以及物理装置无法立刻改变它的运动均会造成延时问题。例如,除了减速以及加速阶段以外,惯性运动的方向是无法立刻变向的,机器人总是需要这样一种过渡阶段。由于这些延迟,机器人执行的动作没有实时的效果,一般在几个时间步骤后就会观察到。与此相反的是,在大多数的强化学习算法中,预设的动作通常都会假定实时起作用。

18.2.4 模型错误灾难

一种抵消真实场景互动成本的方法就是使用精确的模型作为模拟器。在理想环境下,这种方法是能够在仿真模拟下学习到相应的动作,并且随后迁移至真实的机器人。不幸的是,创建一个精确有效的机器人模型以及相应环境是十分困难的。由于小的模型错误可能会积累下来,因此我们频繁看到模拟机器人快速偏离真实系统。当一个训练策略用了不精确的前向

模型作为模拟器时,对于类似 [Atkeson, 1994] 所实验的欠驱动的上摆 (underactuated swing-up) 学习问题,不进行重大修改是无法迁移动作的。作者只在少数实验中,例如在 18.6.3 节的样例中,实现了直接的迁移。如果任务是能量吸收或者消耗能量不太严重的话,那么就可以安全地假设这些应用于仿真模拟中的方法在真实场景下也会很有效 [Kober and Peters, 2010]。在能量吸收的场景下,任务将会十分稳定,并且迁移策略对机器人带来负面作用的风险很低。然而,在能量吸收情境下,由于机械部件之间进行着复杂的交互,任务通常在真实场景下比仿真场景下要学习得更好。在能量发射场景下,任务将会十分不稳定,并且迁移策略将会产生较高的风险。接下来我们将看到,模型最好用来在仿真模拟中测试算法性能,但也可以用来检查距离理论上最优解的差距,或者执行“智力预演”(mental rehearsal)。

[584]

18.2.5 目标规范灾难

在强化学习中,任务的目标是由奖励来隐式地指定。在机器人强化学习中定义一个好的奖励函数通常是一个艰巨的任务。只有在任务完成时才会给予任务奖励,比如说做了一个乒乓球机器人赢得了比赛,将会获得一个简单的二元奖励。然而,机器人通常很少接受这样的奖励,导致真实场景下机器人毕其一生也很难成功。所以,除了只使用简单的二元奖励,我们还需要频繁将额外知识转化为标量奖励来指导学习过程导向一个合理的结果。不同因素之间的平衡是十分重要的,就像打乒乓球时速度非常快将会导致分数很高,但同时也可能会摧毁机器人。好的强化学习算法通常以意想不到的方式使用奖励函数,尤其是当强化学习是应用到局部而非应用到全局的时候。例如,在球拍与球之间接触,就可以作为球运动轨迹的部分奖励(见图 18.2),许多局部最优解都会试图简单地让球保持在球拍上。

逆强化学习,同样也称为学徒学习,是一种有前景的能够替代手动制定奖励函数的方法。与强化学习相反的是,它假定奖励函数能够从一组专家示例中进行重建。最近,它已经成功地应用到了各种机器人中,更多信息参见 [Kolter et al, 2007; Abbeel et al, 2008; Coates et al, 2009]。

18.3 机器人强化学习基础

真实场景下的机器人受基本方法的选择影响程度很大。因此,在这一节,我们将会从一个特定的角度来介绍强化学习。正如第 1 章所言,强化学习的目标是找到一个策略 $\pi(s, a)$, 这个策略能够获得最大程度的奖励 $R(s, a)$ 。然而,在真实场景下平均的奖励由于其稳定属性通常来讲比折扣公式更加合适 [Peters et al, 2004]。为了更好地探索,我们提出了一个条件概率分布的策略 $\pi(s, a) = f(a|s, \theta)$, 参数为 θ 。强化学习的目标在于找到可能的策略 π^* 或者是根据等价策略中的参数 θ^* 使平均奖励 $j(\pi) = \sum_{s,a} \mu^\pi(s) \pi(s,a) R(s,a)$ 最大,其中当策略 π 导致 $T(s, a, s') = P(s'|s, a)$ 时, μ^π 表现为一种稳态分布。因此我们有了以下优化问题

[585]

$$\max_{\mu^\pi, \pi} J(\pi) = \sum_{s,a} \mu^\pi(s) \pi(s,a) R(s,a) \quad (18.1)$$

$$\text{s.t. } \mu^\pi(s') = \sum_{s,a} \mu^\pi(s) \pi(s,a) T(s,a,s'), \forall s' \in S \quad (18.2)$$

$$1 = \sum_{s,a} \mu^\pi(s) \pi(s,a) \quad (18.3)$$

其中,公式 (18.2) 定义了稳态分布 μ^π (例如,它保证了该分布是收敛的) 以及公式 (18.3) 确定了正常状态概率分布。这个优化问题可以通过两种完全不同的方法求解 [Bellman, 1967, 1971], 即,我们可以从最初原始的问题直接搜索最优解,并且我们可以在对偶函数

中优化。在原始公式中的优化称为强化学习中的策略搜索，而在对偶中搜索称为基于价值函数的方法。

18.3.1 价值函数方法

大多数的强化学习的注意力都不会放在直接解决优化问题上，而是解决优化问题的对偶形式 (dual form)。通过使用拉格朗日因子 $V(s')$ 和 \bar{R} ，我们可以通过以下式子来表达这个拉格朗日问题：

$$L = \sum_{s,a} \mu^\pi(s) \pi(s,a) \left[R(s,a) + \sum_{s'} V(s') T(s,a,s') - \bar{R} \right] - \sum_{s'} V(s') \mu^\pi(s') + \bar{R}$$

通过使用直接的结论以及稳定性条件 $\sum_{s',a'} V(s') \mu^\pi(s') \pi(s',a') = \sum_{s,a} V(s) \mu^\pi(s) \pi(s,a)$ ，我们可以通过对 $\mu^\pi(s) \pi(s,a)$ 求导获得库恩塔克 (Karush-Kuhn-Tucker, KKT) 条件 [Kuhn and Tucker, 1950]:

$$\partial_{\mu^\pi} L = R(s,a) + \sum_{s'} V(s') T(s,a,s') - \bar{R} - V(s) = 0$$

这意味着我们拥有的方程数是状态数乘以动作数，但是可以肯定只有一个操作 a^* 是最佳的。因此，我们有以下贝尔曼最优性原则 [Kirk, 1970]:

$$V(s) = \max_{a^*} \left[R(s,a^*) - \bar{R} + \sum_{s'} V(s') T(s,a^*,s') \right] \tag{18.4}$$

586

当我们评估公式 (18.4) 时，我们意识到 $V(s)$ 对应的是奖励差分的总和，差分是状态 s 下取最优行为 a^* 时的值，它的被减数是平均奖励 \bar{R} 。注意，这个函数通常来讲是由人类观察发现的 [Sutton and Barto, 1998]。这个最优性原则不仅创造了最优控制领域，而且从强化学习的角度提出了针对动态规划的解决方案。

因此，对于原始问题我们有了一个对偶函数作为最优性条件。许多传统的强化学习方法都是基于这个方程，这些方法也称为基于价值函数的方法。除了直接学习策略，它们首先对拉格朗日乘子 $V(s)$ 进行近似，也称为价值函数，通过使用价值函数来重构最优策略。大量已存在的方法大致可以分为三类：1) 基于动态规划的最优控制方法，例如策略迭代或者值迭代方法；2) 基于展开的蒙特卡罗算法；3) 时序差分方法，例如 TD(λ)、Q 学习以及 SARSA 算法。详见第 1 章。然而，像这样的基于价值函数的方法通常无法很好地移植到高维度的机器人上，因为对价值函数来说，合适的表示方式本身就很棘手，而找到一个最优解就更是挑战。一个特别重要的问题就是价值函数中由于策略上的一点小变动而导致的误差传播可能会导致价值函数出现巨大改变，而这将会反过来造成策略上很大的改变。

虽然这可能促使更快出现好的结果，可能是全局的最优解，但像这样的学习过程应用到真正的系统时可能会造成严重的破坏，因此它可能会更加危险。表 18.1 概括了使用了基于价值函数方法的论文。在这里，基于模型的方法是指所有应用了预先确定或者已学习好的模型的方法。

表 18.1 展示了已经部署到机器人任务和相关应用中基于价值函数的不同强化学习方法

价值函数方法	
方法	应用到何处
基于模型	Abbeel et al (2006, 2007); Atkeson and Schaal (1997); Atkeson (1998); Bagnell and Schneider (2001); Bakker et al (2006); Coates et al (2009); Donnart and Meyer (1996); Hester et al (2010); Kalmár et al (1998); Ko et al (2007); Kolter et al (2008); Martínez-Marín and Duckett (2005); Michels et al (2005); Morimoto and Doya (2001); Ng et al (2004b,a); Pendrith (1999); Schaal and Atkeson (1994); Touzet (1997); Willgoss and Iqbal (1999)

(续)

价值函数方法	
方法	应用到何处
无模型	Asada et al (1996); Bakker et al (2003); Benbrahim et al (1992); Benbrahim and Franklin (1997); Birdwell and Livingston (2007); Bitzer et al (2010); Conn and Peters II (2007); Duan et al (2007, 2008); Fagg et al (1998); Gaskett et al (2000); Gr " ave et al (2010); Hafner and Riedmiller (2007); Huang and Weng (2002); Ilg et al (1999); Katz et al (2008); Kimura et al (2001); Kirchner (1997); Kroemer et al (2009, 2010); Latzke et al (2007); Lizotte et al (2007); Mahadevan and Connell (1992); Mataric (1997); Nemec et al (2009, 2010); Oßwald et al (2010); Paletta et al (2007); Platt et al (2006); Riedmiller et al (2009); Rottmann et al (2007); Smart and Kaelbling (1998, 2002); Soni and Singh (2006); Thrun (1995); Tokic et al (2009); Uchibe et al (1998); Wang et al (2006)

587

18.3.2 策略搜索

可以很清楚地看到原始公式 (primal formulation) 具有很多针对机器人的特征。它允许专家知识的自然整合, 比如说通过策略的初始化。它允许在不改变原始问题的基础上以合适的形式表达策略的领域内先验结构。最优策略通常具有比最优价值函数少得多的参数, 例如, 在线性二次控制中, 价值函数有二次方个参数, 而策略只需要许多线性比例的参数。很直接地扩展状态与动作空间到连续空间。在策略空间的局部搜索可以直接得到好的结果, 正如早期的登山方法展示的那样 [Kirk, 1970]。额外的限制可以非常自然地并入到策略中, 因此, 策略搜索对于机器人来说更加自然。

然而, 长久以来策略搜索一直被认为是非常困难的问题, 因为最优解无法直接从等式中解析。尽管贝尔曼最优性原则可以从问题的 KKT 条件直接导出。虽然, 在机器人领域, 相比与基于价值函数的方法, 策略搜索在近阶段已经成为了一个重要的选择, 但由于上述原因以及近似价值函数方法的收敛问题, 大多数策略搜索方法通过计算策略的变化 $\delta\pi_i$, 围绕局部存在的策略 π_i 进行优化, 这将增加预期的回报并得到以下形式的迭代更新:

$$\pi_{i+1} = \pi_i + \delta\pi_i$$

这里策略更新的计算方式是关键步骤, 针对更新提出了大量方法, 包括采用有限策略差分 [Geng et al, 2006; Mitsunaga et al, 2005; Sato et al, 2002; Tedrake et al, 2005] 在梯度估计之上的成对比较法 [Strens and Moore, 2001; Ng et al, 2004a], 以及启发式并行搜索方法 (如遗传算法, 参见 [Goldberg, 1989]), 还有来自最优控制的方法, 如差分动态规划 (DDP) [Atkeson, 1998] 以及多重射击方法 [Betts, 2001] 和核心强化学习方法。

最近几年, 强化学习领域已经产生了三种策略搜索方法, 并且这些方法已经很好地移植到了机器人领域: 1) 基于似然比估计的策略梯度方法 [Sutton et al, 2000], 2) 受期望最大化启发的策略更新 [Toussaint et al, 2010], 3) 路径积分法 [Kappen, 2005]。似然比策略梯度法依赖于干扰电机命令而不是策略空间的比较。最初的方法如 REINFORCE [Williams, 1992] 收敛相当缓慢, 但是最新的自然策略梯度方法 [Peters and Schaal, 2008c, b] 已经能够允许更快的收敛, 这可能对机器人来说更加有用。当奖励被视为不合理的概率分布时 [Dayan and Hinton, 1997], 受到期望最大化的启发, 可以推导出安全快速的方法。已经证明这些方法中的一些在机器人中是成功的。例如奖励加权回归 [Peters and Schaal, 2008a]、PoWER [Kober and Peters, 2009]、MCEM [Vlassis et al, 2009] 以及成本正则化核回归 [Kober et al, 2010]。路径积分方法产生的策略更新或多或少与 PoWER (例如 PI2 [Theodorou et al,

2010]) 息息相关，这对于机器人领域是非常有前途的。表 18.2 展示了使用策略搜索方法的论文的概述。

表 18.2 展示了已经应用到机器人任务和相关论文中的基于策略搜索的不同强化学习方法

策略搜索	
方法	应用到何处
梯度	Deisenroth and Rasmussen (2010); Endo et al (2008); Geng et al (2006); Guenter et al (2007); Gullapalli et al (1994); Hailu and Sommer (1998); Kohl and Stone (2004); Kolter and Ng (2009); Mitsunaga et al (2005); Miyamoto et al (1996); Peters and Schaal (2008c,b); Tamei and Shibata (2009); Tedrake (2004); Tedrake et al (2005)
启发式	Erden and Leblebicioaglu (2008); Dorigo and Colombetti (1993); Mataric (1994); Svinin et al (2001); Yasuda and Ohkura (2008); Youssef (2005)
采样	Buchli et al (2011); Kober and Peters (2009); Kober et al (2010); Pastor et al (2011); Peters and Schaal (2008a); Peters et al (2010a)

18.4 表示法带来的可行性

大部分强化学习方法的成功是由于聪明地使用了近似的表达方式。在一个无法做到完全扁平化表示的领域，像这样的近似表示是十分需要的。在机器人中，让强化学习方法变得可行的不同方式都需要与底层框架紧密地耦合。策略搜索方法需要选择一种策略表示来限制可表示策略的数量以达到加速学习，详见 18.4.3 节。基于价值函数的方法需要一个精确、鲁棒但是通用的函数近似，这个函数近似可以充分精确地捕获价值函数，详见 18.4.2 节。通过智能状态 - 动作离散化来减少状态或动作的维度是一种颇具代表性的简化方式，这种简化可以加强策略搜索方法和基于价值函数的方法，详见 18.4.1 节。在表 18.3 中概括了使用表示法来使学习问题变为可行的论文。

表 18.3 展示了采用合适的表示法使机器人强化学习变为可行的各种方法

智能状态 - 动作离散化	
方法	应用到
手工	Benbrahim et al (1992); Kimura et al (2001); Nemec et al (2010);Paletta et al (2007); Tokic et al (2009); Willgoss and Iqbal (1999)
学习	Piater et al (2010); Yasuda and Ohkura (2008)
元动作	Asada et al (1996); Dorigo and Colombetti (1993); Kalmár et al (1998); Mataric (1994, 1997); Platt et al (2006); Soni and Singh (2006); Nemec et al (2009)
关系表示	Cocora et al (2006); Katz et al (2008)
函数近似	
方法	应用到
局部模型	Bentivegna (2004); Schaal (1997); Smart and Kaelbling (1998)
神经网络	Benbrahim and Franklin (1997); Duan et al (2008); Gaskett et al (2000); Hafner and Riedmiller (2003); Riedmiller et al (2009); Thrun (1995)
高斯过程回归	Gräve et al (2010); Kroemer et al (2009, 2010); Lizotte et al (2007); Rottmann et al (2007)
近邻	Hester et al (2010); Mahadevan and Connell (1992); Touzet (1997)
先验结构策略	
方法	应用到
运动原语	Kohl and Stone (2004); Kober and Peters (2009); Peters and Schaal (2008c,b); Theodorou et al (2010)

588
?
589

(续)

先验结构策略	
方法	应用到
神经网络	Endo et al (2008); Geng et al (2006); Gullapalli et al (1994); Hailu and Sommer (1998)
点方法	Miyamoto et al (1996)
线性模型	Tamei and Shibata (2009)
高斯混合模型和局部线性模型 (GMM&LLM)	Deisenroth and Rasmussen (2010); Guenter et al (2007); Peters and Schaal (2008a)
控制器	Kolter and Ng (2009); Tedrake (2004); Tedrake et al (2005); Vlassis et al (2009)
非参数	Kober et al (2010); Mitsunaga et al (2005); Peters et al (2010a)

18.4.1 智能状态 – 动作离散化

具有较低维度的状态或者动作显著地减轻了大多数强化学习问题的难度，特别是在机器人的相关问题上。在这里，我们将会快速地介绍试图通过智能离散化实现这一目标的不同尝试。

手工。大量的作者都会手动调整离散化，以便在真实机器人上学习基本任务。对于低维度的任务，例如在平衡球任务中 [Benbrahim et al, 1992]，我们可以直接生成离散结果，但是更加复杂的任务则需要更多的人类经验。这些任务的范围从具有噪声传感器的基本导航任务 [Willgoss and Iqbal, 1999] 到只有一个自由度的杯中球任务 [Nemec et al, 2010]，两自由度的爬行运动 [Tokic et al, 2009] 以及学习对象的可行性 [Paletta et al, 2007] 直到步行模式变为四条腿行走 [Kimura et al, 2001]。

从数据中学习。除了手动的指定离散化，还可以从数据中学习离散化。例如，基于规则的强化学习方法能自动地分割状态空间以学习移动机器人相关的协作任务 [Yasuda and Ohkura, 2008]。在计算机视觉的相关领域，[Piater et al, 2010] 提出了一种将感知空间自适应并递增地进行离散化，并最终得到离散状态的方法。

元动作。自动构建元动作已经吸引了强化学习研究者的兴趣，并且在文献中有着各种例子。例如在 [Asada et al, 1996] 中，状态与动作集合以某种方式构建，这种方式的特点就是重复的动作原语导致状态的改变，这种改变能够克服与离散化相关联的问题。Q 学习和基于动态规划的方法已经在使用模块的 pick-n-place 任务 [Kalm'ar et al, 1998] 中进行了比较。通过半自动发现的选择可以学习用机器狗运输球的任务 [Soni and Singh, 2006]。类人机器人可以通过仅仅使用原始动作的子目标做到学习浇注任务 [Nemec et al, 2009]。还有各种其他的实例包括搜索 [Mataric, 1997] 和多个机器人之间的合作任务 [Mataric, 1994]、在受限搜索空间中进行抓取操作 [Platt et al, 2006] 以及移动机器人的导航 [Dorigo and Colombetti, 1993]。这些方法属于第 9 章中详细介绍的分层强化学习方法。

关系表示。在关系表示中，状态、动作和转换并不是单独表示的，而是将相同预定义类型的实体分组并且需要考虑它们的关系。这种表示已用来在监督学习 [Cocora et al, 2006] 下对真实的机器人进行建筑物中的导航以及在仿真模拟中操纵关节对象 [Katz et al, 2008]。

18.4.2 函数近似

函数近似一直是允许价值函数方法扩展到感兴趣领域的关键部分。在机器人强化学习

中, 以下函数近似方案已经流行并成功了。

神经网络。神经网络作为用于连续状态和动作的函数逼近器已经被各个研究组使用。例如, 多层感知机用于学习漫游动作和视觉伺服 [Gaskett et al, 2000]; 模糊神经网络 [Duan et al, 2008] 以及基于解释的神经网络 [Thrun, 1995] 允许学习基本的导航任务, 而 CMAC 神经网络已用于两足动物运动 [Benbrahim and Franklin, 1997]。一个特别令人印象深刻的应用是 Brainstormers Robocup 足球队使用多层感知机赢得了几次不同的世界杯联赛 [Hafner and Riedmiller, 2003; Riedmiller et al, 2009]。

近邻单元的扩展。由于神经网络受到局部误差的全局影响, 许多工作都集中在从近邻单元进行简单的扩展。机器人强化学习中最早的文章之一 [Mahadevan and Connell, 1992] 阐述了这样一个思想, 那就是通过统计聚类来加速完成一个移动机器人的推箱子任务, 见图 18.1a。这种方法也用于移动机器人的导航和障碍物躲避等任务 [Touzet, 1997]。类似地, 决策树已用于生成不可见的状态和动作, 例如, 让类人机器人学习罚球 [Hester et al, 2010]。

局部模型。已经证实局部加权回归是一个特别有效的函数逼近器。使用局部模型作为价值函数逼近器可以学习具有障碍物躲避的导航任务 [Smart and Kaelbling, 1998]、杆平衡任务 [Schaal, 1997] 以及空中曲棍球任务 [Bentivegna, 2004]。

高斯过程回归。使用高斯过程回归作为价值函数的函数逼近器, 可以允许自动飞艇学会自主悬停 [Rottmann et al, 2007], 见图 18.1c。类似地, 另一篇论文表明, 可以使用高斯过程回归来学习抓握 [Gräve et al, 2010]。抓握位置利用奖励方式通过高斯过程回归学习得到, 在此过程中尝试那些高奖励预测候选 [Kroemer et al, 2009]。高额奖励的不确定性允许在基于奖励的抓握任务中进行智能化探索 [Kroemer et al, 2010]。机器狗的步态优化可以先通过使用高斯过程回归学习期望奖励函数, 随后从中搜索最优解 [Lizotte et al, 2007]。

18.4.3 预构建策略

[592] 为了使策略搜索方法变为可行, 需要将策略以合适的函数近似形式表示。

运动原语。运动原语是一种受到生物学启发的概念, 它代表基本运动。对于离散运动的动力系统, 运动原语 [Ijspeert et al, 2003; Schaal et al, 2007] 的表现已经应用到学习 T-Ball 的击球任务 [Peters and Schaal, 2008c, b] 和欠驱动摆起以及杯中球任务 [Kober and Peters, 2009], 还有扳动灯开关 [Buchli et al, 2011] 以及打台球和推盒子 [Pastor et al, 2011] 等任务。对于有节奏的动作, 半椭圆机动形式已用来作为机器狗的步态模式 [Kohl and Stone, 2004]。

神经网络。除了解析地描述有节奏感的运动, 神经网络还可以作为振荡器来学习双足机器人的步态 [Geng et al, 2006; Endo et al, 2008]。同样地, 轴孔任务 [见图 18.1b] 和平衡球任务 [Gullapalli et al, 1994] 以及导航任务 [Hailu and Sommer, 1998] 都使用了神经网络作为策略函数逼近器。

点方法。通过点方法优化位置和定时, [Miyamoto et al, 1996] 学习了一个剑道任务。

线性模型。[Tamei and Shibata, 2009] 使用强化学习来调整从 EMG 信号映射到力的模型, 从而用于支持协作握持任务。

高斯混合模型和局部线性模型。其中最一般的函数近似就是基于径向基函数, 也称为高斯核函数。然而确定它们的中心和宽度是具有挑战性的。这些位置和方差可以从强化学习过程前的数据中估计 [Guenter et al, 2007], 并且这个学习过程扩展到伸手运动 [Guenter et al, 2007] 和学习手推杆任务 [Deisenroth and Rasmussen, 2010]。操作空间的控制由 [Peters and

Schaal, 2008a] 使用局部线性模型学习。

控制器。这里需要学习的是局部线性控制器的参数。相关应用包括使用双足机器人在 20 分钟内保持行走 [Tedrake, 2004; Tedrake et al, 2005]。还有如图 18.3 所示, 通过无线控制 (RC) 方式驱动汽车和机器狗的跳跃动作 [Kolter and Ng, 2009] 以及平衡双轮机器人的任务 [Vlassis et al, 2009]。

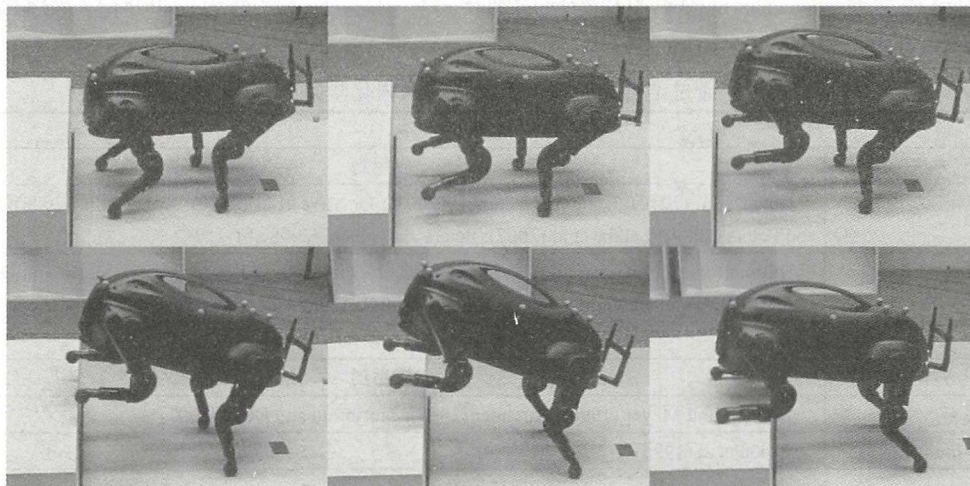


图 18.3 波士顿动力的 LittleDog 在跳跃 [Kolter and Ng, 2009] (图片转载经过了 Zico Kolter 的允许)

非参数。在以下的情况中, 可以用非参数表示。例如, 可以优化不同机器人的相互作用的可能性权重 [Mitsunaga et al, 2005]、乒乓球任务中不同打击运动的权重 [Peters et al, 2010a] 以及用于飞镖和乒乓球任务的元动作的参数 [Kober et al, 2010]。

18.5 先验知识带来的可行性

先验知识可以显著指导学习过程。先验知识包括初始策略、初始模型或者任务预定义结构等形式。这些方法显著地缩小了搜索空间, 从而加快了学习过程。实现初始策略的目标需要允许强化学习方法能够快速探索价值函数或者策略空间中的可能区域, 详见 18.5.1 节。在 18.5.2 节中, 介绍了在任务预构造过程中将复杂任务分解为几个更易处理的任务是非常成功的。表 18-4 中所展示的是使用先验知识使学习问题变为可行的论文的概述。

18.5.1 示范中的先验知识

动物和人经常从模仿试验以及错误中学习。例如, 当学习打网球时, 教练通常给学生做示范如何做适当的挥动, 比如正手或者反手。学生随后将会模仿这种动作, 但是仍然需要数小时的练习才能成功地将球击回对手的区域。因此来自于教练的输入不限于初始的指令, 教练还可以在之后的学习阶段提供额外的示范 [Latzke et al, 2007], 这些也可以用于差分反馈当中 [Argall et al, 2008]。但是全局的搜索是不必要的, 因为学生可以通过局部优化他之前通过模仿得到的打击动作来改进。类似的方法可以加快基于人体演示的强化学习方法或初始手工编码的策略。对于最近的机器人模仿学习综述, 详见 [Argall et al, 2009]。

老师的示范。示范可以通过远程控制机器人来获得, 这种方法也可用于初始化用于

593
594

导航任务的 Q 表 [Connand Peters II, 2007]。如果机器人是反向驱动的, 动作示范 [即可以通过手动的移动] 是可以应用上的。这种方法已经应用到了包括 T-ball 击球 [Peters and Schaal, 2008c, b]、伸手任务 [Guenther et al, 2007; Bitzer et al 2010]、杯中球任务 [Kober and Peters, 2009]、扳动灯开关 [Buchli et al, 2011] 和打台球以及推箱子任务 [Pastor et al, 2011] 等应用。运动捕获装置可以选择使用, 但是由于通信问题, 示范往往不能提供足够信息。通过运动捕获装置获得的示范已经应用到学习杯中球 [Kober et al, 2008] 以及抓握任务中 [Grave et al, 2010]。

表 18.4 展示了通过结合先验知识来使机器人强化学习变为可行的不同方法

示范	
方法	应用到
老师	Bitzer et al (2010); Conn and Peters II (2007); Gr...ave et al (2010); Kober et al (2008); Kober and Peters (2009); Latzke et al (2007); Peters and Schaal (2008c,b)
策略	Birdwell and Livingston (2007); Erden and Leblebicioaglu (2008); Martínez-Marín and Duckett (2005); Smart and Kaelbling (1998); Tedrake (2004); Tedrake et al (2005); Wang et al (2006)
任务架构	
方法	应用到
分层	Donnart and Meyer (1996); Kirchner (1997); Morimoto and Doya (2001)
逐步推进的任务	Asada et al (1996)
指导探索	
方法	应用到
	Huang and Weng (2002); Kroemer et al (2010); Pendrith (1999)

手工编码的策略。预先编写的策略可以提供示范, 而不用人类教师。基于视觉的移动机器人的对接任务使用这种基本的方法比单独使用 [Martínez-Marín and Duckett, 2005] 中描述的 Q 学习方法要明显学习得更快。当机器人明显偏离期望的动作时, 作为备选的校正动作可以用来作为先验知识。这种方法已用于帮助机器狗的行走模式适应新的地面。通过手工编码形成的稳定的初始步态已经可以明显帮助六足机器人 [Erden and Leblebicioaglu, 2008] 以及两足机器人锻炼稳定的步态 [Tedrake, 2004; Tedrake et al, 2005]。

18.5.2 任务结构中的先验知识

通常, 任务可以分层分解为基本组件 (参见第 9 章), 或者是由一系列逐渐增加难度的小型任务组成。在这两种情况下, 学习任务的复杂性都会明显降低。

分层强化学习。更容易的任务可以用作更复杂动作的组件块, 例如, 分层 Q 学习已用于学习六足机器人的不同动作水平: 移动单腿、局部移动完整的身体以及将整个机器人向目标移动 [Kirchner, 1997]。以一个站立任务来说, 本身作为分层强化学习任务, 其中上层使用 Q 学习, 下层使用 TD 学习 [Morimoto and Doya, 2001]。而在迷宫中的导航可以通过调整不同控制模块的影响并且使用演员 - 评论家架构来学习这些模块 [Donnart and Meyer, 1996]。

逐步推进的任务。如果简单的任务已经可以完成的话, 通常复杂的任务将会更容易学习。一系列越来越困难的复杂目标通过使用 Q 学习方法应用于学习目标任务中 [Asada et al, 1996]。

18.5.3 先验知识指导探索

移动机器人学习引导注意 [Huang and Weng, 2002] 是通过采用一种新颖的改进后的 Q 学习方法，这种方法使用了“矫正的截断回报”并考虑到估计方差，使用步进反射的六足机器人可以学习如何走路 [Pendrith, 1999]。使用高信度限制直接的探索抓握可以有效地学习如何抓握 [Kroemer et al, 2010]。离线搜索可以用于在抓握任务期间增强 Q 学习的效果 [Wang et al, 2006]。

18.6 仿真模拟带来的可行性

使用仿真模拟替代真实的物理机器人具有诸如安全性和速度的重要优点。模拟可以消除明显不良的表现，并且通常来讲运行得比真实的更快。模拟无疑是调试算法的有用的测试平台。一种流行的方法是结合仿真与真实机器人的各自优势，在真实系统中测试有前途的策略以收集新的数据来重新细化模拟细节（见 18.6.2 节）。不幸的是，直接将在模拟中学习到的策略应用到真实系统中是极具挑战性的（详见 18.6.3 节）。表 18.5 中展示了使用仿真模拟来使学习问题变得可行的论文概述。

596

表 18.5 该表展示出了使用仿真模拟来使机器人强化学习变为可行的不同方法

仿真模拟	
方法	应用到
智力预演	Abbeel et al (2006); Atkeson and Schaal (1997); Atkeson et al (1997); Atkeson (1998); Bagnell and Schneider (2001); Bakker et al (2006); Coates et al (2009); Deisenroth and Rasmussen (2010); Ko et al (2007); Kolter et al (2008); Michels et al (2005); Nemec et al (2010); Ng et al (2004b,a); Schaal and Atkeson (1994); Uchibe et al (1998)
直接的策略迁移	Bakker et al (2003); Duan et al (2007); Fagg et al (1998); Ilg et al (1999); Oßwald et al (2010); Svinin et al (2001); Youssef (2005)



图 18.4 自主反转直升机飞行 [Ng et al, 2004b] (图片转载经由 Andrew Ng 许可)

18.6.1 模型的作用

无模型算法尝试直接学习价值函数或策略。基于模型的方法综合了学习系统的模型以及价值函数或策略。基于模型的方法可以使学习过程中有效处理更多的样本。然而，根据模型类型的不同，有可能需要占据大量的内存。基于模型的方法依赖于能在模型中找到好的策略的方法，这些方法有可能面临模型不准确的风险。如果学习方法要求预测未来或者使用了

导数,那么这种误差就有可能快速地累积,因此有可能显著放大噪声和错误。这些问题导致价值函数或者策略在模型中工作良好,然而在真实系统中却表现很差。这个问题与 18.2.4 节中讨论到的迁移问题息息相关。关于这个问题的解决方案就是放大噪声影响,引入一定量的不一致性 [Atkeson, 1998], 或者使用粗略模型来找到一种策略能够补偿真实系统中动作的导数。有关更详细的讨论,请参见第 4 章。

[597] 在 18.2.4 节中,我们讨论了在模拟中学习的策略经常不能直接迁移到真实系统中。然而,仿真模拟仍然是一种非常有用的工具。大多数仿真的运行速度要明显比真实场景下更快,并且可以避免昂贵实验中出现的相关问题(见 18.2.2 节)。由于这些原因,仿真模拟通常用于调试、测试和优化算法。在仿真模拟中学习通常比真实机器人要容易得多。因为可以控制噪声并且所有变量都可以访问。如果这种方法在仿真模拟中无法工作的话,它通常不太可能在真实系统上直接有效工作。许多论文同样使用仿真模拟作为评估方法的依据,因为在真实系统上频繁地重复试验来观察平均的表现以及比较许多算法的性能通常是不可行的。

18.6.2 智力预演

将仿真中的学习和真实场景下的学习结合起来的想法因为强化学习中的 Dyna 架构 [Sutton, 1990] 流行起来。由于它在机器人领域中的显著优点,本文中也提及这个想法。现实世界中收集到的经验可以用于从数据中学习出一个前向模型 [Åström and Wittenmark, 1989]。这种前向模型允许在模拟环境中进行训练,随后将产生的策略迁移到真实场景下。这种方法也可以被迭代,并且能够显著降低与真实场景之间的交互。然而,通常学习过程可以利用模型误差,但是这可能会导致结果的偏差以及收敛变慢。

在强化学习中像这样的智力预演已经出现在了許多应用中。在仿真和定向搜索的并行学习中允许 Q 学习在 20 分钟内从头开始学习导航任务 [Bakker et al, 2006]。两个机器人轮流学习简单的足球任务也可以从智力预演中获益 [Uchibe et al, 1998]。Atkeson et al, 1997] 使用了前向模型学习一个桌球任务和魔鬼棍任务。[Nemec et al, 2010] 在仿真中使用了一个学习好的价值函数来初始化真实的机器人学习。

为了减少由模型误差引起的仿真偏差, [Ng et al, 2004b, a] 建议在学习的模拟器中重新使用一系列随机数用于机器人,这个建议被称为 PEGASUS。注意,这种有着固定模型的方法在仿真社区 [Glynn, 1987] 中是众所周知的。这种方法已经广泛应用在自动直升机学习高级技术动作的任务中 [Bagnell and Schneider, 2001; Ng et al, 2004b, a], 见图 18.4。同样的方法还应用于学习遥控车的控制参数 [Michels et al, 2005] 以及自动飞艇中 [Ko et al, 2007]。在 [Abbeel et al, 2006] 中已经提出了在现实世界的数据中使用基本策略的粗糙模型的替代方法,并且这种方法已经用于学习控制遥控车的转向。除了基于前向模型的仿真器采样以外,像这样的学习模型还可直接用于计算最优控制策略。这导致产生了各种机器人强化学习的应用,例如,使用 DDP 学习的摆锤摆起任务 [Atkeson and Schaal, 1997; Atkeson, 1998]、基于局部 LQR 解的恶魔棍(一种陀螺仪形式的杂技) [Schaal and Atkeson, 1994]、在自动遥控车上使用 DDP 方法训练一个执行轨迹跟踪的空间索引控制器 [Kolter et al, 2008]、手推杆任务 [Deisenroth and Rasmussen, 2010]、使用 DDP 训练的特技直升机飞行 [Coates et al, 2009]。使用多个概率模型解决 LQR 问题并将得到的闭环控制与开环控制相结合,导致自主侧向滑动到停车点 [Kolter et al, 2010]。一个有前景的相关新方法就是 LQR 树 [Tedrake et al, 2010]。

[598]

18.6.3 从仿真直接迁移到真实机器人

只有少数论文声称，在仿真中学习到的策略可以直接迁移到真实的机器人当中并保持其高性能。例如，移动机器人的迷宫导航任务 [Bakker et al, 2003; Oßwald et al, 2010; Youssef, 2005] 和移动机器人的障碍物躲避任务 [Fagg et al, 1998]。在非常基本的机器人足球 [Duan et al, 2007] 和多腿机器人中实现了类似的迁移 [Ilg et al, 1999; Svinin et al, 2001]。

18.7 一个学习样例：杯中球任务

在此之前，我们已经审查了大量的问题和相关的机器人强化学习的解决方案。这里，我们将会采用截然不同的方法详细讨论之前研究过的一个任务。这个任务称为“杯中球任务”，由于其复杂性，它可以作为一个突出各种挑战和方法的样例来详细讨论。在 18.7.1 节中，我们描述了以任务和奖励为重点的实验设置。在 18.7.2 节中讨论了一种在机器人中特别有用的预结构化策略。在 18.7.3 节中提出了先验知识。在 18.7.4 节中解释了所采用的策略搜索算法的优点。在 18.7.5 节中，我们将会讨论在任务中使用仿真模拟，在 18.7.6 节中，我们将会探讨一种替代的强化学习方法。

18.7.1 实验设置：任务和奖励

儿童的电机游戏杯中球，也称为 balero 和 bilboquet，即使对于大多数成年人来说也是挑战。玩具包括一只手握的小杯子（或者，在我们的例子中，它被连接到机器人的末端执行器上），一个小球悬挂在一个连接到杯子底部的绳上（我们的玩具绳长 40cm）。最初球是静止的，保持竖直的状态，玩家需要快速通过绳子牵引球导致球发生运动，并试图用杯子接住。

599

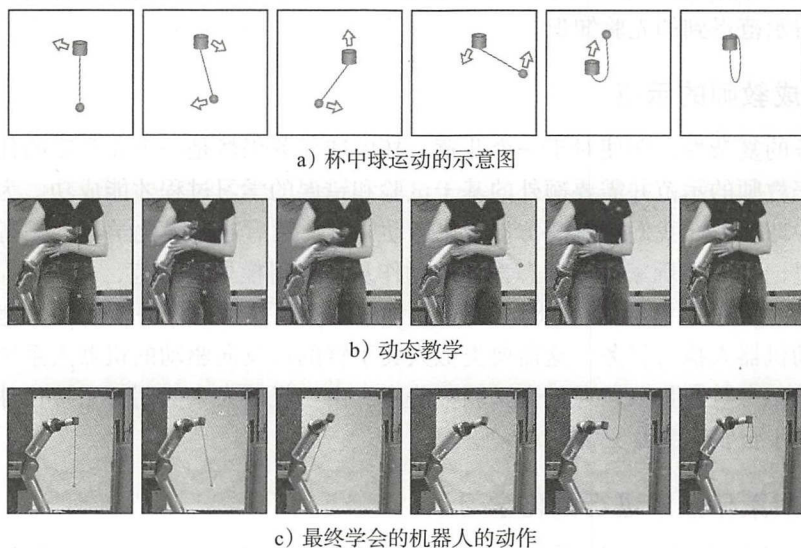


图 18.5 该图展示了 a) 杯中球运动时的示意图，b) 动态教学过程，c) 最终学会动作的机器人的运动。箭头显示了该帧中当前运动的方向。人类的杯子的运动通过模拟学习迁移到机器人，学习过程中每个关节都有 31 个参数，并进行大约 3 秒的运动。机器人设法准确地再现模仿的运动，但是球由于几厘米的误差错过了杯子。在通过使用基于加权探索奖励策略学习 (PoWER) 算法迭代大约 75 次之后，机器人已经能改进它的动作使球经常进入到杯中。在图 18-6 中可以看到性能的提升

图 18.5a 中展示出了可能的移动。由于绳子经常会发生缠绕从而导致失败而机器人无法解开它,所以需要人为干预,并且自动复位通常来讲是不可能的。系统的状态可以由机器人的关节角度和关节速度以及球的笛卡尔坐标和速度来描述。动作就是关节空间加速度(其通过反向动态控制器可以转换成扭矩)。因此,我们必须处理二十种状态和七个动作维度。基于价值函数的方法对状态-动作空间进行离散化是几乎不可行的。

在时间点 t_c , 球以向下的方向通过杯子的边缘, 对于所有的 $t \neq t_c$, $r(t)=0$ 我们计算奖励公式为 $r(t_c)=\exp(-\alpha(x_c-x_b)^2-\alpha(y_c-y_b)^2)$ 。在这里, 杯子的位置是由 $[x_c, y_c, z_c] \in \mathbb{R}^3$ 表示, 球的位置由 $[x_b, y_b, z_b] \in \mathbb{R}^3$ 决定, 并且我们使用的缩放参数 $\alpha=100$ 。开始时, 我们仅使用基于球与杯子之间最小距离的奖励函数。然而, 该算法会偏向于利用从下面或侧面用球击杯子, 因为这样的动作更容易实现并产生相对高的奖励。为了避免这种局部最优, 如最初描述的那样必须先找到一个良好的奖励函数。

600

这个任务是相当复杂的, 因为奖励不仅受到杯子的运动影响, 而且最重要的是球的运动。因为球的运动非常敏感, 只要受到小的扰动、初始条件或者小臂运动变化, 都会严重影响结果。由于绳子存在非线性和不可观察的动态性以及绳子自身不可忽略的重量等原因, 创建一个准确的仿真模拟是非常困难的。

18.7.2 适当的策略表示

该策略由动态系统运动原语来表示 [Ijspeert et al, 2003; Schaal et al, 2007]。全局运动被编码为点吸引子线性动态系统。移动的细节由一个允许学习复杂动作的转换函数生成。这个转换函数使用局部线性函数近似地建模。全局吸引动作和局部转换的组合允许非常简约地表示策略。这个策略在参数 $a=\theta\mu(s)$ 时是线性的, 因此可以方便地包含来自使用局部加权回归的监督学习示范得到的先验知识。

18.7.3 生成教师的示范

由于任务的复杂性, 即使对于一个儿童, 杯中球任务仍然是一个非常难的任务, 也只能在刚刚观察完教师的示范并需要额外的基于试验和错误的学习过程才能成功。为了模仿儿童如何学习杯中球任务, 我们首先模仿初始化运动原型, 然后通过强化学习来提高。

我们通过记录人类玩家进行动态教学的动作从而获得模仿的示范, 如图 18-5b 所示。动态教学意味着“用手拿机器人”, 此时处于重力补偿模式并且记录关节角度、速度和加速度时, 通过移动机器人执行任务。这需要类似人类手臂的可反向驱动的机器人系统。从模仿开始, 所需的策略参数数量可以通过交叉验证确定。由于机器人未能用杯子接住球, 所以额外的强化学习对于自我完善是必需的。

18.7.4 使用策略搜索进行强化学习

当学习运动原语时, 我们学习了一个确定的均值策略 $\bar{a}=\theta\mu(s)=f(z)$, 这个策略对于参数 θ 是线性的, 并且加上探索 $\varepsilon(s,t)$ 增强使得非参数强化学习成为可能。结果就是探索策略可以用 $a=\theta\mu(s,t)+\varepsilon(\mu(s,t))$ 的形式给出。策略搜索方法通常会聚焦于状态无关的高斯探索, 即 $\varepsilon(\mu(s,t)) \sim \mathcal{N}(0, \Sigma)$, 就用到如 T-Ball 棒球游戏 [Peters and Schaal, 2008c] 和受限运动 [Guenther et al, 2007] 等应用中。然而, 以我们的经验来看, 这种无结构的探索每一步都有几个缺点, 即, 1) 它会造成大的方差, 并且会随着时间步变大而增长; 2) 它会过于频繁地干

601

扰动动作, 因此, 会“冲淡”动作本身的作用; 3) 执行轨迹的过程中会损害系统。

或者, 人们也可以选择生成有结构的、状态依赖的探索方案 [Ruckstie et al, 2008], $\varepsilon(\mu(s, t)) = \varepsilon_t(\mu(s, t), [\varepsilon_t]_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2))$, 其中 σ_{ij}^2 是探索过程的元参数, 它也可以被优化。这一参数引出的策略 $a \sim \pi(a_t | s_t, t) = \mathcal{N}(a | \mu(s, t), \hat{\Sigma}(s, t))$ 。

在 [Kober and Peters, 2009] 中, 我们推导了一个奖励加权模仿的框架。基于 [Dayan and Hinton, 1997], 我们把每次奖励的回报作为一个不正确的概率分布。我们最大化期望回报的对数下界。归功于优化下界和探索策略, 这个框架产出了几个著名的策略搜索算法: 回合 REINFORCE [Williams, 1992]、策略梯度理论 [Sutton et al, 2000]、回合的自然演员-评论家算法 [Peters and Schaal, 2008b]、广义回报加权回归 [Peters and Schaal, 2008a] 以及我们提出的通过加权探索回报的策略学习 (Policy learning by Weighting Exploration with Returns, PoWER) 新算法。PoWER 是一个受期望最大化所启发的算法, 它使用状态依赖探索。更新规则是:

$$\theta' = \theta + \frac{E_r \left\{ \sum_{t=1}^T \varepsilon_r Q^\pi(s_t, a_t, t) \right\}}{E_r \left\{ \sum_{t=1}^T Q^\pi(s_t, a_t, t) \right\}}$$

在这个策略为基础的情境下, 为了减小尝试的次数, 我们通过重要性采样 [Sutton and Barto, 1998] 来重用尝试。为了避免由重要性采样引发的强化学习中的脆弱性, 非常小的重要性权重会被丢弃。这个算法基本上执行了一个围绕先验知识的局部搜索。

图 18.6 表示在多次学习后的期望回报。图中可以清晰地看出何处收敛到最大值。大概 75 次尝试之后, 机器人通常可以把球放入杯中。

使用一个基于价值函数的方法需要大量的样本来获得对价值函数的一个好估计。在如此高维的运行空间中贪婪地搜索一个最优运动命令可能与找到一个局部最优策略一样困难。

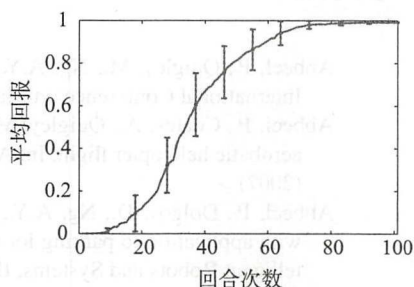


图 18.6 显示了杯中球实验中, 在 20 次测试上予以平均, 得到已学习的策略的期望回报值

18.7.5 机器人强化学习中使用仿真模拟

我们通过利用数据学习参数的刚体动力学创造了一种机器人的模拟方法。玩具被模拟为一个具有弹性绳的摆, 当小球比绳子离杯子更近并且绳子很长时它就变成一个飞行质点。通过调整弹簧、阻尼和恢复常数来匹配记录的数据。即使这种模拟非常符合记录的数据, 抓球入杯的模拟常常会在真实环境中错失杯子几厘米, 反过来也一样。然而, 这种模拟对于开发和调试程序是很有用的, 因为它在模拟中比实际中跑得更快, 并且也不需要人类的监督或者干预。

18.7.6 价值函数方法的替代方案

[Nemec et al, 2010] 使用了一个不同的强化学习方法来使用 Mitsubishi PA10 机器人完成杯中球任务。他们把这个任务分解成两个子任务——上摆阶段和抓取阶段。在上摆阶段, 球被移动到杯子的上端。在抓取阶段, 通过对飞行质点的移动轨迹的解析预测来抓球。抓取

动作是固定的, 仅仅学习上摆动作。这篇论文提出使用 SARSA 来学习上摆过程。状态由点的位置和速度以及角位置和角速度组成。机器人的运动通过调节单一笛卡尔坐标系方向的加速度实现。通过对状态 (324 个值) 和动作 (5 个值) 的离散化, 以及使用模拟方法进行初始化来实现简化。这种动作最初是通过模拟 220 到 300 次来进行学习的。状态-动作价值函数用来初始化真实机器人的学习过程。在真实环境中机器人需要额外进行 40 到 90 次来适应来自模拟状态下的动作。

18.8 总结

603 本章中, 我们综述了机器人强化学习, 来向通用强化学习领域的读者们介绍这一领域最先进的技术。我们已经指明了一些其中的问题比如高维连续状态-动作空间、尝试过程中关联的高代价问题、从仿真到真实机器人迁移策略问题以及对于适宜的奖励函数的需求。对于机器人强化学习方法之间有多大差异的讨论受到领域的影响。我们已经综述了不同作者的方法, 从而通过改良的表示、加入先验知识以及使用模拟方法来使得机器人强化学习容易进行。为了强调我们认为重要的内容, 给出了一个关于机器人如何学习如杯中球这样复杂任务的示例。

参考文献

- Abbeel, P., Quigley, M., Ng, A.Y.: Using inaccurate models in reinforcement learning. In: International Conference on Machine Learning, ICML (2006)
- Abbeel, P., Coates, A., Quigley, M., Ng, A.Y.: An application of reinforcement learning to aerobatic helicopter flight. In: Advances in Neural Information Processing Systems, NIPS (2007)
- Abbeel, P., Dolgov, D., Ng, A.Y., Thrun, S.: Apprenticeship learning for motion planning with application to parking lot navigation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2008)
- Argall, B.D., Browning, B., Veloso, M.: Learning robot motion control with demonstration and advice-operators. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2008)
- Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. *Robotics and Autonomous Systems* 57, 469–483 (2009)
- Asada, M., Noda, S., Tawaratsumida, S., Hosoda, K.: Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine Learning* 23(2-3), 279–303 (1996)
- Atkeson, C., Moore, A., Stefan, S.: Locally weighted learning for control. *AI Review* 11, 75–113 (1997)
- Atkeson, C.G.: Using local trajectory optimizers to speed up global optimization in dynamic programming. In: Advances in Neural Information Processing Systems, NIPS (1994)
- Atkeson, C.G.: Nonparametric model-based reinforcement learning. In: Advances in Neural Information Processing Systems, NIPS (1998)
- Atkeson, C.G., Schaal, S.: Robot learning from demonstration. In: International Conference on Machine Learning, ICML (1997)
- Bagnell, J.A., Schneider, J.C.: Autonomous helicopter control using reinforcement learning policy search methods. In: IEEE International Conference on Robotics and Automation, ICRA (2001)
- Bakker, B., Zhumatiy, V., Gruener, G., Schmidhuber, J.: A robot that reinforcement-learns to identify and memorize important previous observations. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2003)
- Bakker, B., Zhumatiy, V., Gruener, G., Schmidhuber, J.: Quasi-online reinforcement learning

- for robots. In: IEEE International Conference on Robotics and Automation, ICRA (2006)
- Barto, A.G., Mahadevan, S.: Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems* 13(4), 341–379 (2003)
- Bellman, R.E.: *Dynamic Programming*. Princeton University Press, Princeton (1957)
- Bellman, R.E.: *Introduction to the Mathematical Theory of Control Processes*, vol. 40-I. Academic Press, New York (1967)
- Bellman, R.E.: *Introduction to the Mathematical Theory of Control Processes*, vol. 40-II. Academic Press, New York (1971)
- Benbrahim, H., Franklin, J.A.: Biped dynamic walking using reinforcement learning. *Robotics and Autonomous Systems* 22(3-4), 283–302 (1997)
- Benbrahim, H., Doleac, J., Franklin, J., Selfridge, O.: Real-time learning: a ball on a beam. In: *International Joint Conference on Neural Networks, IJCNN* (1992)
- Bentivegna, D.C.: *Learning from observation using primitives*. PhD thesis, Georgia Institute of Technology (2004)
- Betts, J.T.: Practical methods for optimal control using nonlinear programming. In: *Advances in Design and Control*, vol. 3. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (2001)
- Birdwell, N., Livingston, S.: Reinforcement learning in sensor-guided aibo robots. Tech. rep., University of Tennessee, Knoxville, advised by Dr. Itamar Elhanany (2007)
- Bitzer, S., Howard, M., Vijayakumar, S.: Using dimensionality reduction to exploit constraints in reinforcement learning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS* (2010)
- Buchli, J., Stulp, F., Theodorou, E., Schaal, S.: Learning variable impedance control. *International Journal of Robotics Research Online First* (2011)
- Buşoniu, L., Babuška, R., De Schutter, B., Ernst, D.: *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, Boca Raton (2010)
- Coates, A., Abbeel, P., Ng, A.Y.: Apprenticeship learning for helicopter control. *Commun. ACM* 52(7), 97–105 (2009)
- Cocora, A., Kersting, K., Plagemann, C., Burgard, W., Raedt, L.D.: Learning relational navigation policies. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS* (2006)
- Conn, K., Peters II, R.A.: Reinforcement learning with a supervisor for a mobile robot in a real-world environment. In: *IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA* (2007)
- Dayan, P., Hinton, G.E.: Using expectation-maximization for reinforcement learning. *Neural Computation* 9(2), 271–278 (1997)
- Deisenroth, M.P., Rasmussen, C.E.: A practical and conceptual framework for learning in control. Tech. Rep. UW-CSE-10-06-01, Department of Computer Science & Engineering, University of Washington, USA (2010)
- Donnart, J.Y., Meyer, J.A.: Learning reactive and planning rules in a motivationally autonomous animat. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 26(3), 381–395 (1996)
- Dorigo, M., Colombetti, M.: Robot shaping: Developing situated agents through learning. Tech. rep., International Computer Science Institute, Berkeley, CA (1993)
- Duan, Y., Liu, Q., Xu, X.: Application of reinforcement learning in robot soccer. *Engineering Applications of Artificial Intelligence* 20(7), 936–950 (2007)
- Duan, Y., Cui, B., Yang, H.: Robot Navigation Based on Fuzzy RL Algorithm. In: Sun, F., Zhang, J., Tan, Y., Cao, J., Yu, W. (eds.) *ISNN 2008, Part I. LNCS*, vol. 5263, pp. 391–399. Springer, Heidelberg (2008)
- Endo, G., Morimoto, J., Matsubara, T., Nakanishi, J., Cheng, G.: Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot. *I. J. Robotic Res.* 27(2), 213–228 (2008)
- Erden, M.S., Leblebicioğlu, K.: Free gait generation with reinforcement learning for a six-legged robot. *Robot. Auton. Syst.* 56(3), 199–212 (2008)
- Fagg, A.H., Lotspeich, D.L., Hoff, J., Bekey, G.A.: Rapid reinforcement learning for reactive control policy design for autonomous robots. In: *Artificial Life in Robotics* (1998)

- Gaskett, C., Fletcher, L., Zelinsky, A.: Reinforcement learning for a vision based mobile robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2000)
- Geng, T., Porr, B., Wörgötter, F.: Fast biped walking with a reflexive controller and real-time policy searching. In: Advances in Neural Information Processing Systems, NIPS (2006)
- Glynn, P.: Likelihood ratio gradient estimation: an overview. In: Winter Simulation Conference, WSC (1987)
- Goldberg, D.E.: Genetic algorithms. Addison Wesley (1989)
- Gräve, K., Stückler, J., Behnke, S.: Learning motion skills from expert demonstrations and own experience using gaussian process regression. In: Joint International Symposium on Robotics (ISR) and German Conference on Robotics, ROBOTIK (2010)
- Guenther, F., Hersch, M., Calinon, S., Billard, A.: Reinforcement learning for imitating constrained reaching movements. *Advanced Robotics* 21(13), 1521–1544 (2007)
- Gullapalli, V., Franklin, J., Benbrahim, H.: Acquiring robot skills via reinforcement learning. *IEEE on Control Systems Magazine* 14(1), 13–24 (1994)
- Hafner, R., Riedmiller, M.: Reinforcement learning on a omnidirectional mobile robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2003)
- Hafner, R., Riedmiller, M.: Neural reinforcement learning controllers for a real robot application. In: IEEE International Conference on Robotics and Automation, ICRA (2007)
- Hailu, G., Sommer, G.: Integrating symbolic knowledge in reinforcement learning. In: IEEE International Conference on Systems, Man and Cybernetics (SMC) (1998)
- Hester, T., Quinlan, M., Stone, P.: Generalized model learning for reinforcement learning on a humanoid robot. In: IEEE International Conference on Robotics and Automation, ICRA (2010)
- Huang, X., Weng, J.: Novelty and reinforcement learning in the value system of developmental robots. In: Lund University Cognitive Studies (2002)
- Ijspeert, A.J., Nakanishi, J., Schaal, S.: Learning attractor landscapes for learning motor primitives. In: Advances in Neural Information Processing Systems, NIPS (2003)
- Ilg, W., Albiez, J., Jedele, H., Berns, K., Dillmann, R.: Adaptive periodic movement control for the four legged walking machine BISAM. In: IEEE International Conference on Robotics and Automation, ICRA (1999)
- Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4, 237–285 (1996)
- Kalmár, Z., Szepesvári, C., Lörincz, A.: Modular Reinforcement Learning: An Application to a Real Robot Task. In: Birk, A., Demiris, J. (eds.) *EWLR 1997. LNCS (LNAI)*, vol. 1545, pp. 29–45. Springer, Heidelberg (1998)
- Kappen, H.: Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment* 11 (2005)
- Katz, D., Pyuro, Y., Brock, O.: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation. In: *Robotics: Science and Systems, R:SS* (2008)
- Kimura, H., Yamashita, T., Kobayashi, S.: Reinforcement learning of walking behavior for a four-legged robot. In: *IEEE Conference on Decision and Control (CDC)* (2001)
- Kirchner, F.: Q-learning of complex behaviours on a six-legged walking machine. In: *EUROMICRO Workshop on Advanced Mobile Robots* (1997)
- Kirk, D.E.: *Optimal control theory*. Prentice-Hall, Englewood Cliffs (1970)
- Ko, J., Klein, D.J., Fox, D., Hähnel, D.: Gaussian processes and reinforcement learning for identification and control of an autonomous blimp. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2007)
- Kober, J., Peters, J.: Policy search for motor primitives in robotics. In: *Advances in Neural Information Processing Systems, NIPS* (2009)
- Kober, J., Peters, J.: Policy search for motor primitives in robotics. *Machine Learning Online First* (2010)
- Kober, J., Mohler, B., Peters, J.: Learning perceptual coupling for motor primitives. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS* (2008)

- Kober, J., Oztop, E., Peters, J.: Reinforcement learning to adjust robot movements to new situations. In: *Robotics: Science and Systems Conference (R:SS)* (2010)
- Kohl, N., Stone, P.: Policy gradient reinforcement learning for fast quadrupedal locomotion. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2004)
- Kolter, J.Z., Ng, A.Y.: Policy search via the signed derivative. In: *Robotics: Science and Systems (R:SS)* (2009)
- Kolter, J.Z., Abbeel, P., Ng, A.Y.: Hierarchical apprenticeship learning with application to quadruped locomotion. In: *Advances in Neural Information Processing Systems (NIPS)* (2007)
- Kolter, J.Z., Coates, A., Ng, A.Y., Gu, Y., DuHadway, C.: Space-indexed dynamic programming: learning to follow trajectories. In: *International Conference on Machine Learning (ICML)* (2008)
- Kolter, J.Z., Plagemann, C., Jackson, D.T., Ng, A.Y., Thrun, S.: A probabilistic approach to mixed open-loop and closed-loop control, with application to extreme autonomous driving. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2010)
- Kroemer, O., Detry, R., Piater, J., Peters, J.: Active learning using mean shift optimization for robot grasping. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2009)
- Kroemer, O., Detry, R., Piater, J., Peters, J.: Combining active learning and reactive control for robot grasping. *Robotics and Autonomous Systems* 58(9), 1105–1116 (2010)
- Kuhn, H.W., Tucker, A.W.: Nonlinear programming. In: *Berkeley Symposium on Mathematical Statistics and Probability* (1950)
- Latzke, T., Behnke, S., Bennewitz, M.: Imitative Reinforcement Learning for Soccer Playing Robots. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) *RoboCup 2006. LNCS (LNAI)*, vol. 4434, pp. 47–58. Springer, Heidelberg (2007)
- Lizotte, D., Wang, T., Bowling, M., Schuurmans, D.: Automatic gait optimization with gaussian process regression. In: *International Joint Conference on Artificial Intelligence (IJCAI)* (2007)
- Mahadevan, S., Connell, J.: Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence* 55(2-3), 311–365 (1992)
- Martínez-Marín, T., Duckett, T.: Fast reinforcement learning for vision-guided mobile robots. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2005)
- Mataric, M.J.: Reward functions for accelerated learning. In: *International Conference on Machine Learning (ICML)* (1994)
- Mataric, M.J.: Reinforcement learning in the multi-robot domain. *Autonomous Robots* 4, 73–83 (1997)
- Michels, J., Saxena, A., Ng, A.Y.: High speed obstacle avoidance using monocular vision and reinforcement learning. In: *International Conference on Machine Learning (ICML)* (2005)
- Mitsunaga, N., Smith, C., Kanda, T., Ishiguro, H., Hagita, N.: Robot behavior adaptation for human-robot interaction based on policy gradient reinforcement learning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2005)
- Miyamoto, H., Schaal, S., Gandolfo, F., Gomi, H., Koike, Y., Osu, R., Nakano, E., Wada, Y., Kawato, M.: A kendama learning robot based on bi-directional theory. *Neural Networks* 9(8), 1281–1302 (1996)
- Morimoto, J., Doya, K.: Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems* 36(1), 37–51 (2001)
- Nakanishi, J., Cory, R., Mistry, M., Peters, J., Schaal, S.: Operational space control: a theoretical and empirical comparison. *International Journal of Robotics Research* 27, 737–757 (2008)
- Nemec, B., Tamošiūnaitė, M., Wörgötter, F., Ude, A.: Task adaptation through exploration and action sequencing. In: *IEEE-RAS International Conference on Humanoid Robots, Humanoids* (2009)
- Nemec, B., Zorko, M., Zlajpah, L.: Learning of a ball-in-a-cup playing robot. In: *International Workshop on Robotics in Alpe-Adria-Danube Region (RAAD)* (2010)
- Ng, A.Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., Liang, E.:

- Autonomous inverted helicopter flight via reinforcement learning. In: International Symposium on Experimental Robotics (ISER) (2004a)
- Ng, A.Y., Kim, H.J., Jordan, M.I., Sastry, S.: Autonomous helicopter flight via reinforcement learning. In: Advances in Neural Information Processing Systems (NIPS) (2004b)
- Oßwald, S., Hornung, A., Bennewitz, M.: Learning reliable and efficient navigation with a humanoid. In: IEEE International Conference on Robotics and Automation (ICRA) (2010)
- Paletta, L., Fritz, G., Kintzler, F., Irran, J., Dorffner, G.: Perception and Developmental Learning of Affordances in Autonomous Robots. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI 2007. LNCS (LNAI), vol. 4667, pp. 235–250. Springer, Heidelberg (2007)
- Pastor, P., Kalakrishnan, M., Chitta, S., Theodorou, E., Schaal, S.: Skill learning and task outcome prediction for manipulation. In: IEEE International Conference on Robotics and Automation (ICRA) (2011)
- Pendrith, M.: Reinforcement learning in situated agents: Some theoretical problems and practical solutions. In: European Workshop on Learning Robots (EWRL) (1999)
- Peters, J., Schaal, S.: Learning to control in operational space. *International Journal of Robotics Research* 27(2), 197–212 (2008a)
- Peters, J., Schaal, S.: Natural actor-critic. *Neurocomputing* 71(7-9), 1180–1190 (2008b)
- Peters, J., Schaal, S.: Reinforcement learning of motor skills with policy gradients. *Neural Networks* 21(4), 682–697 (2008c)
- Peters, J., Vijayakumar, S., Schaal, S.: Linear quadratic regulation as benchmark for policy gradient methods. Tech. rep., University of Southern California (2004)
- Peters, J., Mülling, K., Altun, Y.: Relative entropy policy search. In: National Conference on Artificial Intelligence (AAAI) (2010a)
- Peters, J., Mülling, K., Kober, J., Nguyen-Tuong, D., Kroemer, O.: Towards motor skill learning for robotics. In: International Symposium on Robotics Research, ISRR (2010b)
- Piater, J., Jodogne, S., Detry, R., Kraft, D., Krüger, N., Kroemer, O., Peters, J.: Learning visual representations for perception-action systems. *International Journal of Robotics Research Online First* (2010)
- Platt, R., Grunewald, R.A., Fagg, A.H.: Improving grasp skills using schema structured learning. In: International Conference on Development and Learning (2006)
- Åström, K.J., Wittenmark, B.: Adaptive control. Addison-Wesley, Reading (1989)
- Riedmiller, M., Gabel, T., Hafner, R., Lange, S.: Reinforcement learning for robot soccer. *Autonomous Robots* 27(1), 55–73 (2009)
- Rottmann, A., Plagemann, C., Hilgers, P., Burgard, W.: Autonomous blimp control using model-free reinforcement learning in a continuous state and action space. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2007)
- Rückstieß, T., Felder, M., Schmidhuber, J.: State-Dependent Exploration for Policy Gradient Methods. In: Daellemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 234–249. Springer, Heidelberg (2008)
- Sato, M.-A., Nakamura, Y., Ishii, S.: Reinforcement Learning for Biped Locomotion. In: Dorronsoro, J.R. (ed.) ICANN 2002. LNCS, vol. 2415, pp. 777–782. Springer, Heidelberg (2002)
- Schaal, S.: Learning from demonstration. In: Advances in Neural Information Processing Systems, NIPS (1997)
- Schaal, S., Atkeson, C.G.: Robot juggling: An implementation of memory-based learning. *Control Systems Magazine* 14(1), 57–71 (1994)
- Schaal, S., Atkeson, C.G., Vijayakumar, S.: Scalable techniques from nonparametric statistics for real-time robot learning. *Applied Intelligence* 17(1), 49–60 (2002)
- Schaal, S., Mohajerian, P., Ijspeert, A.J.: Dynamics systems vs. optimal control - a unifying view. *Progress in Brain Research* 165(1), 425–445 (2007)
- Smart, W.D., Kaelbling, L.P.: A framework for reinforcement learning on real robots. In: National Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence, AAAI/IAAI (1998)
- Smart, W.D., Kaelbling, L.P.: Effective reinforcement learning for mobile robots. In: IEEE International Conference on Robotics and Automation (ICRA) (2002)
- Soni, V., Singh, S.: Reinforcement learning of hierarchical skills on the sony aibo robot. In:

- International Conference on Development and Learning (ICDL) (2006)
- Strens, M., Moore, A.: Direct policy search using paired statistical tests. In: International Conference on Machine Learning (ICML) (2001)
- Sutton, R., Barto, A.: Reinforcement Learning. MIT Press, Boston (1998)
- Sutton, R.S.: Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In: International Machine Learning Conference (1990)
- Sutton, R.S., McAllester, D., Singh, S., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: Advances in Neural Information Processing Systems (NIPS) (2000)
- Sutton, R.S., Koop, A., Silver, D.: On the role of tracking in stationary environments. In: International Conference on Machine Learning (ICML) (2007)
- Svinin, M.M., Yamada, K., Ueda, K.: Emergent synthesis of motion patterns for locomotion robots. Artificial Intelligence in Engineering 15(4), 353–363 (2001)
- Tamei, T., Shibata, T.: Policy Gradient Learning of Cooperative Interaction with a Robot Using User's Biological Signals. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008. LNCS, vol. 5507, pp. 1029–1037. Springer, Heidelberg (2009)
- Tedrake, R.: Stochastic policy gradient reinforcement learning on a simple 3d biped. In: International Conference on Intelligent Robots and Systems (IROS) (2004)
- Tedrake, R., Zhang, T.W., Seung, H.S.: Learning to walk in 20 minutes. In: Yale Workshop on Adaptive and Learning Systems (2005)
- Tedrake, R., Manchester, I.R., Tobenkin, M.M., Roberts, J.W.: LQR-trees: Feedback motion planning via sums of squares verification. International Journal of Robotics Research 29, 1038–1052 (2010)
- Theodorou, E.A., Buchli, J., Schaal, S.: Reinforcement learning of motor skills in high dimensions: A path integral approach. In: IEEE International Conference on Robotics and Automation (ICRA) (2010)
- Thrun, S.: An approach to learning mobile robot navigation. Robotics and Autonomous Systems 15, 301–319 (1995)
- Tokic, M., Ertel, W., Fessler, J.: The crawler, a class room demonstrator for reinforcement learning. In: International Florida Artificial Intelligence Research Society Conference (FLAIRS) (2009)
- Toussaint, M., Storkey, A., Harmeling, S.: Expectation-Maximization methods for solving (PO)MDPs and optimal control problems. In: Inference and Learning in Dynamic Models. Cambridge University Press (2010)
- Touzet, C.: Neural reinforcement learning for behaviour synthesis. Robotics and Autonomous Systems, Special Issue on Learning Robot: the New Wave 22(3-4), 251–281 (1997)
- Uchibe, E., Asada, M., Hosoda, K.: Cooperative behavior acquisition in multi mobile robots environment by reinforcement learning based on state vector estimation. In: IEEE International Conference on Robotics and Automation (ICRA) (1998)
- Vlassis, N., Toussaint, M., Kontes, G., Piperidis, S.: Learning model-free robot control by a Monte Carlo EM algorithm. Autonomous Robots 27(2), 123–130 (2009)
- Wang, B., Li, J., Liu, H.: A heuristic reinforcement learning for robot approaching objects. In: IEEE Conference on Robotics, Automation and Mechatronics (2006)
- Willgoss, R.A., Iqbal, J.: Reinforcement learning of behaviors in mobile robots using noisy infrared sensing. In: Australian Conference on Robotics and Automation (1999)
- Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning 8, 229–256 (1992)
- Yasuda, T., Ohkura, K.: A Reinforcement Learning Technique with an Adaptive Action Generator for a Multi-Robot System. In: Asada, M., Hallam, J.C.T., Meyer, J.-A., Tani, J. (eds.) SAB 2008. LNCS (LNAI), vol. 5040, pp. 250–259. Springer, Heidelberg (2008)
- Youssef, S.M.: Neuro-based learning of mobile robots with evolutionary path planning. In: ICGST International Conference on Automation, Robotics and Autonomous Systems (ARAS) (2005)

第六部分

Reinforcement Learning: State-of-the-Art

结 束 语

本书内容提要

总结、未来方向和展望

Marco Wiering, Martijn van Otterlo

19.1 回顾

本书对强化学习 (Reinforcement Learning, RL) 领域进行了深入而完整的描述。在本书最后一个章节我们会首先讨论本书已经覆盖的内容。之后描述书中没有包含的主题, 没有对它们展开讨论主要是因为它们不属于强化学习的主要领域或者它们只是强化学习中一个很小的子领域 (可能是新颖的或者新兴的领域)。在回顾完强化学习和这本书之后, 我们会讨论强化学习领域未来的发展趋势, 最后, 作为全书的结束, 我们征集了本书部分作者的观点, 列出他们认为的未来最重要的研究主题。

19.1.1 本书覆盖内容

613 本书一直把描述强化学习领域的主体内容作为目标。它集中讨论了强化学习领域的两本重要经典书籍 [Sutton et al, 1998 ; Bertsekas and Tsitsiklis, 1996] 中未被讨论的所有研究大方向。从本书第 2 章描述的强化学习领域最著名的方法, 到中间几章涉及的高效解决方案, 这些方案比 15 年前提出的方案好多了。本书巨大的体量和大量的引用也清楚地表明了这个领域巨大的进步, 其中最重要的突破可能就是 TD 算法 [Sutton, 1988] 和 Q 学习算法 [Watkins, 1989; Watkins and Dayan, 1992] 的发现。很多更新的进展更好地利用了学习器的经验, 比如批处理强化学习 (第 2 章)、最小二乘法的策略迭代 (第 3 章)、模型的运用 (第 4 章) 以及知识迁移 (第 5 章)。第 6 章则分析了更优探索方法的理论优势以获取更好的经验。

本书的第三部分给出了强化学习中各种表示方式的不同用途。表示的方式可以是基于向量的表示 (第 7 章)、使用一阶逻辑的表示 (第 8 章)、有效地运用分层表示 (第 9 章) 或者当使用演化算法时基本上使用的无偏表示 (第 10 章)。当在学习过程中使用了正确的表示方式, 学习就可以变得更有效并且学习到的策略相比而言要智能得多, 正如使用一阶逻辑程序与价值函数时。

在第四部分中, 我们描述了不同的概率框架和算法。第 11 章描述了新颖的贝叶斯强化学习框架。第 12 章介绍了部分可观察的马尔可夫决策过程和高效的解决方案。第 13 章描述了可预测的状态表示, 其中学习器的过往经验被简洁地描述为关于未来的一系列期望。第 14 章, 多学习器的扩展与博弈理论的合作、协调和竞争等概念一并给出。紧接着, 第 15 章描述了去中心化的部分可观察的马尔可夫决策过程框架和为解决这些困难问题的规划算法。本书最终以第 16 ~ 18 章作结, 其中第 16 章围绕背景信息和强化学习与人类学习的关系, 第 17 章探讨强化学习在游戏领域的成功应用, 第 18 章讲述强化学习在机器人控制中的应用。

19.1.2 哪些主题没有被包含

强化学习是一个包含在机器学习领域中的主题。无监督学习和监督学习方法经常被强化学习高效地应用在大型状态-动作空间 (state-action space) 中。本书的目标是描述强化学习近期的进展, 因此通用机器学习的相关内容没有完全涵盖。强化学习技术可以与很多种类的回归方法结合起来, 从与环境有限的交互中学习价值函数。所以回归模型的新进展在强化学习领域同样很重要。我们推荐一些机器学习的书籍, 例如 [Mitchell, 1996; Alpaydin, 2010; Bishop, 2006] 等。这些机器学习书籍可以帮助读者学习完整的机器学习领域, 但大多数机器学习著作中仅仅只有一章介绍强化学习的内容。

除了与机器学习相关以外, 强化学习还与规划算法和控制理论有着同样的目标。这些领域通常使用一个稍许不同的符号系统, 但是都通过一个学习器或者控制器调整自身动作来最优地达到设计目标。我们在本书中未能覆盖这些强化学习领域相近的领域, 因为它们有着更驳杂的背景和更大的研究社区。

614

因此, 我们将会讨论那些没有被包含但是常常被认为是强化学习技术的主题, 虽然它们不需要有价值函数的基础, 但是仍然在诸如演化强化学习 [Moriarty and Miikkulainen, 1996] 的章节中被讨论。尽管有些强化学习的研究方向在这个成果丰富的领域有着令人振奋的新进展, 然而本书没有涉及它们。我们将首先在这里讨论它们, 之后将会讨论一些已经应用强化学习并且有效解决了困难问题的应用领域。

19.1.2.1 强化学习更多的进展

当然, 一本书是无法穷尽强化学习领域的全部内容的, 尤其是在这个领域已经存在 20 多年并且研究社区每年都在扩大的情况下。因此下面所列举的清单是无穷无尽的, 但是它展示了一些有趣的观点, 已经验证这些观点在更高效的强化学习解决特定问题的过程中是有用的。

在线规划与展开方法。在大规模状态-动作空间和有限的资源条件下, 精确地逼近价值函数会变得非常困难。通常情况下函数逼近器是无法完美拟合目标价值函数的, 这会导致选择和执行错误的动作。对于这个问题的部分补救措施是使用前瞻规划来利用决策过程中更多的信息。例如, 这个技术已成功地应用在 TD-Gammon 程序 [Tesauro, 1995] 中, 使得 TD-Gammon 作为一个双陆棋玩家程序更加强大。另外一个有趣的技术是蒙特卡罗搜索树 (Monte Carlo Tree Search, MCTS), 在一个致力于把人工与计算智能应用于博弈程序的研究社区里, 蒙特卡罗搜索树吸引了大量的关注。该技巧使用快速多次模拟的方法来在学习器未来的多条路径中采样结果。MCTS 在围棋游戏中是十分有效的 [Coulom, 2006]。这种方法试图避免直接使用没有精确逼近到最优的价值函数。这些技巧通过平均在多次模拟中产生不同动作的输出, 使它们在应对估计中的不确定性时能够更加健壮。在线规划方法中还有一个技术是基于模型的预测控制 (MPC) 这个技术在控制理论领域中广为人知, 它使用水平移动窗口来计算不同有限动作序列的回报 (return)。获得最多奖励的“规划”将会在第一步中执行, 然后重复这个过程。这些 MPC 技术在确定性任务中是很有用的, 但是在处理很多不确定或者随机输出时可能有一些问题。

好奇心与内部激励。学习器可以从学习子任务序列中获益, 比如在执行使用电脑这一子任务之前需要先执行坐下子任务。学习自适应的子目标有时要用高频访问状态来找到瓶颈状态, 但是这样的方法明显需要大量的经验数据。使用内部激励与创造力来最大化某些生成的

615

奖励函数, 无论是否有详细的先验知识, 都能够提升强化学习 (特别是基于模型的强化学习) 对于世界的理解。使用好奇心与创造力来生成琐碎的、新颖的并且令人惊奇的动作和数据是一个新的进展 [Schmidhuber, 1991b, a, 2010]。这样的算法应该有两个学习组件: 一个通用的奖励优化器或者强化学习器, 与一个对于学习器不断增长的历史数据的自适应编码器。其中学习器的历史数据是指学习器与环境的交互记录。编码器的学习过程是一个不断向奖励优化器提供内部奖励的过程。这就意味着, 后者受到激励来发明新的时空模式, 这些模式编码器虽然并不理解但是能够轻松学习到更好的编码方式。这个过程仅仅消耗少量的计算资源。为了最大化奖励的期望 (在外部奖励缺失的情况下), 奖励优化器将会创造更多更复杂的动作来获得暂时新奇但是长期却乏味的模型来使得编码器快速地进步。

带有自适应状态分割的基于模型的强化学习。在学习器经验的基础上, 基于模型的算法首先估计一个模型, 然后使用动态规划方法计算出一个策略。这种方法的优势是一旦估计出转换模型, 就可能使用新的奖励函数。比如一个奖励函数在进入目标状态时给出 1 的奖励, 而其他状态的奖励都是 0, 这样一个新的策略就能够计算出来。这种知识复用通常是不可能无模型强化学习方法中使用的。虽然基于模型的技术对处理样本很高效, 它们在处理连续状态和动作时还是会出问题, 因为在这样的条件下存储转换概率 (或者概率密度) 是很困难的。解决这个问题一个可能方向是使用自适应状态分割来在一个抽象状态中对相似的状态进行聚类, 以及在给定离散动作下在抽象状态之间估计转换概率。这个途径也许会导致模型不再遵守马尔可夫性, 因此把它们与先前状态和动作的记忆进行合并是有用的, 正如一些解决 POMDP 问题的技术所使用的那样 [Ring, 1994; McCallum, 1995]。

其他强化学习方法与函数逼近技术。有大量的强化学习方法已经在前文提及。例如, [Wiering and van Hasselt, 2009] 提出了五个新的强化学习方法。其他最近提出的方法是双 Q 学习 [van Hasselt, 2010] 和贪婪 Q 学习 [Azar et al, 2011]。通常来说这些算法在特定环境下有一些优势, 但是也许会在其他环境下表现得更差。由于强化学习算法的数量太多, 所以本书没有全部涵盖它们。同样的, 强化学习算法与各种函数逼近器的结合也没能完全覆盖。我们也没有延伸到递归神经网络与强化学习的结合, 比如 [Schmidhuber, 1990; Bakker, Schmidhuber, 2004], 虽然它们对于 POMDP 或者多学习器环境是非常有用的工具。在多学习器强化学习中, 强化学习记忆先前 (与环境) 的交互来对其他学习器进行建模是有益的, 并且可能把学习器引向对于学习器本身很有好处的动作。其他的函数逼近器, 比如支持向量机 [Vapnik, 1995] 已经用在强化学习中 [Dietterich and Wang, 2002], 但是还没有引起强化学习研究社区的兴趣, 因为研究人员们更喜欢使用广义线性模型或者神经网络。

安全探索与实时学习问题。当强化学习直接应用在机器人或者控制其他物理设备上时, 系统可能在使用没有任何先验知识的情况下得到的自适应控制器时受到损害, 而避免这样的损害是非常重要的, 比如从白板学习过程中产生的损害。在训练与人类用户交互的强化学习学习器的实例中, 强化学习学习器探索很多动作的过程不是人类用户期望的, 因此人类用户会因学习器愚蠢的动作而受挫。因此, 在这样的情况下也许有必要把强化学习与事先存在的安全的控制器结合起来。然后可以用强化学习在很多情况下改变控制器的输出, 但是控制器也可以防止学习器改变它生成的动作。

另一个对于实时学习有趣的问题是使用先验知识。许多强化学习研究者已经证明了强化学习可以在不用先验知识的情况下学习。然而, 这种学习方式的代价是它通常需要大量的交互来使学习器取得良好的表现。一些研究通过向学习器展示先验知识来替代从零开始学习,

这些先验知识可能是回报已存在的控制器或者是给学习器一些示范生成的初始动作给出。也可能给出任务的抽象表示（作为先验知识）[van Otterlo, 2003]，因此只有 Q 值或者其他的参数需要被学习。如果这些先验知识很容易得到并且很难被学习器学会，那么使用先验知识是很重要的。

程序演化。强化学习领域已经提出了好多种方法，但是这些方法的实现方式差异很大。其中一类方法是分类器系统。分类系统源于用桶队算法（bucket brigade algorithm）来对学习器过去的条件-动作规则赋予信任值或者惩罚值。分类器系统也与演化算法相结合来进化出一组好的规则，也可能是由于这个原因这种方法更多地演化计算领域被研究。EIRA[Schmidhuber, 1996] 是一个原则，它促使学习器改变其自身的程序并且朝着从环境中获取并积累更多奖励的方向持续这种改变。EIRA 已经与莱文搜索（Levin Search）一起使用 [Wiering and Schmidhuber, 1996; Schmidhuber et al, 1997c]，并且引出了 success-story 算法 [Schmidhuber et al, 1997b, a, c]，这个算法使用自修正策略来获取确定越来越高的奖励。遗传规划 [Cramer, 1985; Koza, 1992, 1994] 与相关技巧如 PIPE[Sakustowicz and Schmidhuber, 1997] 已经用来解决强化学习问题。与分类器系统类似，它们也更多地演化计算社区中研究。

通用学习器与最优问题求解。研究理论最优算法来解决控制问题的论文正在不断增长。虽然是基于差异很大的原理，这方面的进展也十分有趣。柯尔莫哥洛夫复杂性理论（Kolmogorov complexity）[Kolmogorov, 1965; Solomonoff, 1964; Li and Vitányi, 1990] 提供了最优预测和通用归纳推理 [Solomonoff, 1964, 1978] 的基础。理论上最优但是不可计算的强化学习算法 Axi [Hutter, 2004] 已经有了几个实现的程序了 [Poland and Hutter, 2006; Veness et al, 2011]，其中一个例子是玩 PacMan 游戏[⊖]。 [617]

绝大多数的理论分析算法是以寻找解决问题的最短程序为基础的。这类算法的一个问题是需要测试许多程序，通常是按复杂度的顺序进行。假如一个短程序可以计算出最优解，那么这种方法就会非常高效。在这类的研究中已经出现一些进展。首先是速度优先思想，它要求导出的是处理数据的最快的程序而不是最短的程序，并且有关研究已经表明速度优先可以导出一种可计算策略，这种策略能在给定过去数据情况下产生对未来的最优预测。与莱文搜索不同之处在于，这种叫作最优排序问题求解（Optimal Ordered Problem Solving, OOPS）的方法并没有忽略隐藏在算法复杂度计算中巨大的常量。OOPS 在使用经验过程中逐渐减小 O 符号前的常量 [Schmidhuber, 2004]。在很短的几天之内 OOPS 学习到了一个用于解决 n 个盘子的汉诺塔问题的通用求解程序，解决了最高达 $n=30$ 的汉诺塔问题，最短的程序用了多于 10^9 步才解决这个问题（该程序没有使用搜索来解决）。Gödel 机 [Schmidhuber, 2009] 是一种强化学习器或者问题求解器，它具有一流的数学严格性、通用性、全自引用、自我完善和最高效率等特性强化学习。受到 Gödel 著名的自引用公式（1931）的启发，Gödel 机能够重写自己的任意部分的代码，只要它发现这样的重写是有用的。

19.1.2.2 强化学习的不同应用领域

虽然本书已经描述了机器人学和博弈论两大应用领域，但还有很多研究使用强化学习在几个其他有趣的领域中让学习器学习如何动作。下面给出其中的部分领域。

经济应用。在经济应用中，学习器的目标是在适当的时间尽可能多地赚钱。一个例子

⊖ 食豆小子，是 Namco 公司于 1980 年推出的经典游戏。——译者注

是使用强化学习学习器在股票市场中交易股票或者在给定浮动汇率的情况下买卖外国货币 [Nevmyvaka et al, 2006]。这样的问题中预测未来是很困难甚至是不可能的。所以实现的技巧通常需要让学习器从给出的动态变化的历史价格中学习并且最优化收益, 然后使用最好的学习器在近期进行交易。虽然这种方法可能行得通, 但通常难以处理无法预测的事件, 比如股票市场崩溃。另一个经济应用是让学习器在网络上交易物品或者为订票公司规划旅行路线。这样的交易学习器也许有很多好处, 比如相比交易人员和规划人员, 它们速度更快并且成本更低。

网络应用。很多真实问题可以使用网络建模, 在网络中一些项沿着边移动。强化学习在研究网络节点中优化学习器的最早成功应用之一是在因特网上路由传递信息 [Littman and Boyan, 1993; Di Caro and Dorigo, 1998]。在这样的网络中, 数据包抵达网络中的节点, 然后每个节点中的学习器学习最优路径来把数据包导向目标节点。这些自适应方法是很有前景的, 尤其在非常饱和的网络中。与在网络中路由信息相似, 强化学习学习器可以应用在智能电网中来控制能量流动和存储。另一种类型的网络是日常交通网络, 每一辆车都有一个特殊的目的地, 在十字路口(网络中的节点)可以通过操作红绿灯来避免拥堵。在 [Wiering, 2000] 中, 强化学习可用来优化学习器以控制交通网中的红绿灯。结果表明学习器比多种固定控制器表现更好。最后一个应用是用强化学习学习器优化认知无线电和手机频段的使用情况。这个应用中, 学习器的目标是最小化特定频段的呼叫丢失情况。

学习沟通。强化学习学习器可以用来选择自然语言相关的动作。讲话或者书写一些词语可以看作一种特殊的动作。困难在于短自然语句的数量十分庞大, 可能是与现有的巨大的词汇量有关。为了解决这个问题, [Singh et al, 2002] 描述了使用固定数量的语句来回答打电话给计算机询问特定即将发生事件的人。因为计算机使用语音识别来理解人类用户时, 无法做到完全理解呼叫者想表达的信息。因此, 概率框架如 MDP 和 POMDP 可以在这样的任务上大显身手。

组合优化问题。运筹学和元启发式方法是一个很大的领域, 其目的是解决没有必要与学习器交互的复杂问题。这些问题通常使用矩阵表示问题状态, 并且使用代价函数找出最优解。其中一种组合优化解决方案叫作蚁群算法 [Dorigo et al, 1996], 这种方法使用一群蚂蚁来做局部判断, 类似于强化学习中的学习器。每一只蚂蚁执行一系列局部决策促使蚁群产生解决方案。然后使用额外的信息素 (pheromone) 对最优决策序列进行奖励, 奖励将会影响到未来所有蚂蚁的决策。使用这种方法, 任意一个蚁群优化过程都与强化学习很相似。蚁群算法已经成功应用在诸如旅行商问题 [Dorigo and Gambardella, 1997]、二次分配问题 [Gambardella et al, 1999] 和网络通信路由 [Di Caro and Dorigo, 1998] 等问题。其他多学习器强化学习方法也用于组合优化问题当中, 比如作业调度问题和负载均衡问题 [Riedmiller and Riedmiller, 1999]。

实验测试平台。当研究人员开发出一个新的强化学习方法, 他们通常会对这个新方法做实验来测试与之前已有的方法相比怎么样。当一个研究人员比较不同论文不同算法的结果时会出现一个问题, 那就是一般不同论文使用的问题参数会有轻微的区别。例如, 当离散时间步大一点的时候, 山地车问题 (mountain car problem) 会变得更加容易解决, 并且山地车只需要更少的动作次数就可以爬到山顶。出于这个原因, 一些研究人员开发了 RL Glue [Tanner and White, 2009], 有了这个平台就可以在强化学习算法之间举行竞赛。强化学习比赛已经在挑战性游戏如俄罗斯方块 (Tetris) 和马里奥兄弟 (Mario Bros) 上举行。这样的实验测试

平台使得不同方法之间公平比较成为可能，并且这样的标准平台应该可以帮助回答以下问题：哪一种强化学习方法在哪一种环境下表现最好？

19.2 展望未来

强化学习方法吸引了众多的关注，因为它们可以让学习器在各种不同的问题上自动地执行任务。虽然强化学习早已成功地运用在许多玩具问题或者某些真实场景中，为了将强化学习运用到更多感兴趣的问题上，未来仍有很多工作要做。在这一节我们会先关注目前还没有完全解决的研究性问题，然后再检视一些看起来用强化学习方法无法解决的应用问题。

19.2.1 目前未知的内容

我们首先描述一下强化学习中已经很久没有解决的问题，其中很多强化学习未解决的问题在已知其解的情况下可以变得更高效。

什么是最好的函数逼近器？强化学习已经与表格表示法结合来存储价值函数。虽然已证明绝大多数的算法都是收敛的，对于大规模或者连续的状态-动作空间而言，函数逼近器还是必需的。出于这个原因大多数强化学习研究者们使用广义线性模型或者多层感知机，广义线性模型需要选择固定的基函数集合。线性模型中只有基函数的激活值与状态值估计之间的参数必须学习，因此其优点是线性模型速度快并且可以使用线性代数高效计算出估计值。缺点是这一组基函数需要先验地选择，并且使用的基函数通常都是十分局部的函数，例如模糊规则或者径向基函数，而这些函数并不能很好地扩展到输入维度很高的情况。神经网络可以学习隐式特征并且很好地使用数理逻辑激活函数来创造出比局部基函数更短的隐式表达。它们还更加灵活，因为神经网络可以学习基函数的布局。然而训练神经网络需要大量的经验。我们基本上可以使用机器学习中的所有回归算法来表达价值函数。比如，在 [Dietterich and Wang, 2002] 中提出支持向量机 (support vector machine) 与强化学习相结合，而在 [Ernst et al, 2005] 中使用随机森林 (random forest) 进行强化学习。最近，几种特征构造技巧（比如，基于奖励的启发式方法或者贝尔曼残差 (Bellman residual)）就是为强化学习和动态规划算法开发出来的，但是还需要更多的工作来找出在何时以及如何使用它们才能达到最大效果。在 [Sutton and Barto, 1998] 中提出了卡内尔瓦编码 (Kanerva encoding)，这种编码方式可能扩展到高维数据中。以下问题仍有待解决：对于什么样的问题哪些函数逼近器效果最好。

620

通用算法的收敛证明。现在已经有大量工作研究与特定函数逼近器结合的强化学习算法的收敛性。例如，研究广义线性模型结合的强化学习算法的收敛性。在这个方向上的一些研究 [Schoknecht, 2002; Maei et al, 2010] 已经表明在特定假设情况下，一些结合的算法是收敛的。通常这些证明都不能推广到所有控制条件下，只能应用在一个固定策略的价值函数已经被估计好的情况。我们希望看到更多有关保证许多函数逼近器收敛性的证明，以及这些证明所需要的条件有哪些。并且，收敛的速度也很重要。这样的研究能够帮助从业者对于特定的问题选取最佳的方法。

最优探索。许多研究工作集中在借助最优探索来找到最重要的学习经验。然而，哪些探索方法将会对哪些特定问题产生最佳效果还是一个未解决的问题。当使用函数逼近器替代查找表时，如何为一个问题选择最优探索策略的问题就会变得十分困难。一些研究工作 [Nouri and Littman, 2010] 对于连续空间提出了一种高效的探索方法。这样的问题一般都是非常困难

的,因为在训练时间有限的情况下,学习器通常难以访问到环境的每一个部分。因此这里需要做一个权衡。为困难问题创造出最高效的探索策略仍然是强化学习中未发展起来的主题,因此大多数研究人员还是用着简单的 ϵ 贪婪与玻尔兹曼探索来解决他们自己的问题。

可扩展性问题。强化学习早已经在特定的包含很多高维输入的复杂问题上成功运用了。最知名的例子要数西洋双陆棋(TD-Gammon) [Tesauro, 1995] 和电梯调度(Elevator Dispatching) [Crites and Barto, 1996] 了。然而,十几年过去了,对于强化学习扩展到任何控制问题上所需要的条件,我们仍然知之甚少。很有可能是因为强化学习在一些问题上表现良好的时候会在另一些问题上表现糟糕。例如,西洋双陆棋问题的成功在于双陆棋游戏的随机性使得价值函数是平滑的。在机器学习中,使用强化学习逼近平滑的函数比波动起伏的函数要简单得多。当把强化学习应用于解决复杂情况时,加强强化学习对问题的理解是很重要的,因为这将使得强化学习应用到很多工业级应用上,其中一些应用通常正在应用其他方法解决问题,例如使用控制论的方法。

621 **大脑是如何进行强化学习的?** 旧脑(old brain)包含了产生情感和感觉的区域,它们与知觉输入相连接。这些情感从孩提时代开始就处于很重要的地位,并且我们可以使用强化学习的术语来解释它们:为皮质提供奖励。人体中也存在着从额叶皮质到旧脑的反馈循环,因此人类可以思考并且操控他感知的方式。虽然强化学习系统可以大体类比于人脑,我们对于人脑的具体工作机制还是了解得不够多,而这些工作机制能够帮助我们在学习器中实现更多类人的学习能力。很多神经科学的研究目前正在聚焦于人脑更加详尽的工作机制以及更加精确的测量设备,但是我们预计要详细地了解人类学习过程还有很长一段路要走。

19.2.2 看起来不可能的强化学习应用

我们感觉强化学习能够使用到不同的控制任务中,这些任务需要学习器做决策来操纵环境。如下所示,我们给出一些对于强化学习来说十分困难的应用。

仅使用强化学习达到通用人工智能。通用智能需要感知、自然语言处理、执行复杂动作以及更多其他功能。对于不同的问题,不同的人工智能子领域都正在致力于开发出比以前更好的算法。通用人工智能领域试图开发出能够具有上述所有能力的学习器。虽然强化学习可以在每一领域中发挥作用,但仅仅使用强化学习创造上述不同的智能技能并不可取。强化学习能够预测未来也许是智能体最重要的一个能力。当我们看到一辆车在路上行驶,我们能够精确地判断出几秒后这辆车会在哪个地方。当我们下楼梯时我们能够预测什么时候我们的脚能够踩到楼梯。强化学习可以在预测问题中成功地运用,因此在创造通用智能的过程中强化学习会扮演一个重要的角色,但是在缺少其他人工智能子领域的发展下无法做到通用智能。

使用基于价值函数的强化学习实现程序。强化学习的研究人员们通常会面临的一个控制问题就是编程实现一个强化学习学习器程序和一个环境程序。研究自编的强化学习方法是很有意思的,比如 Göedel 机 [Schmidhuber, 2009]。这将允许强化学习学习器能够编程实现更好的学习器,然后产生更好的学习器,这个过程是无穷无尽的。根据我们的观察,现在基于价值函数的强化学习方法不太可能做到这一点的。原因是寻找正确的程序的过程几乎是大海捞针:几乎没有多少程序是有用的,只有极少数程序能够解决问题。不能够从部分奖励中学习的话,强化学习的最优方法是不太可能创造程序的。需要提醒的是已经有高效的强化学习编译器存在了 [McGovern and a. Andrew G. Barto, 1999],但是这个应用使用了大量的领域知识来使它工作良好。通用编程语言中使用强化学习在未来将是一个有趣的研究方向

622

[Simpkins et al, 2008]。这项研究将会使得用所有的知识写程序这一愿景成为可能，并且可以用强化学习优化该程序的其他部分。

19.2.3 有趣的方向

我们将给出一些能够提升强化学习效率的研究方向。我们还会描述一些还没有完全展现出强化学习潜能的应用领域。

更佳世界建模技巧。基于模型的强化学习能够比无模型的强化学习更有效地学习经验。目前有很多模型构建的技巧存在的，比如批处理强化学习 [Riedmiller, 2005; Ernst et al, 2009]、LSPI (见第 3 章)、优先扫除技巧 [Moore and Atkeson, 1993]、Dyna [Sutton, 1990] 以及最佳匹配学习 [van Seijen et al, 2011]。问题是我们是否可以使用向量量化技术 (vector quantization technique) 来离散化连续状态空间，然后创造性地应用类动态规划算法。例如，对于一个机器人导航任务，知名的 SIFT 特征 [Lowe, 2004] 可以用来构造一组视觉关键词。然后多个子状态会在模型中变得活跃，这使规划和表示问题变得棘手。比如通过使用特定的回归算法，可以用上一个时间步激活的关键词来估计视觉关键词在下一个时间步时被激活的概率，并且结合被激活的关键词来估计奖励和价值函数的值。这个研究方向出现了很多有趣的新问题。

高度并行强化学习。随着分布式计算设备的普及，一个有趣的方向是研究强化学习如何最大化利用云计算和超级计算发展成果。在线性模型的例子中，我们就可以研究是否平均线性模型中的权值能够得到比独立 (线性) 模型更好的效果。一些研究工作已经聚焦在强化学习中的集成方法 [Wiering and van Hasselt, 2008]，这些方法可以从分布式计算设备上获益。

结合强化学习与示范学习。在很多应用中控制器与学习器通常已经存在，这就允许强化学习可以先从示范经验中学习，然后强化学习学习器可以对自己进行微调来优化性能。在机器人学中，示范学习早已成功地运用了，如 [Coates et al, 2009; Peters et al, 2003; Peters and Schaal, 2008; Kober and Peters, 2011]。然而在示范学习运用到西洋双陆棋游戏的实例中却没有获得性能的提升 [Wiering, 2010]。因此可以在哪些基于示范学习的领域结合示范学习与强化学习来探索新的问题是颇具挑战性的。

强化学习应用于复杂游戏。在一些非常复杂的游戏，比如围棋中，业内已经可以使用蒙特卡罗搜索树让电脑成为最好的玩家 [Coulom, 2006]。强化学习方法很少能够用在玩这类游戏之中，因为精确地表示和学习价值函数是非常困难的。所以很多搜索和采样方法目前比知识密集型解决方案更有效。在搜索和知识上的权衡 [Berliner, 1997] 表明运用更多的知识需要以减少搜索为代价，因为这样的模型包含更多参数，需要消耗更多的时间求值。理解最新的强化学习方法是有意義的，这些方法可以非常精确地学习逼近复杂游戏的价值函数，并且仍然能够很快地计算结果。一些这个方向上的研究已经在国际象棋上使用 [Baxter et al, 1997; Veness et al, 2009]，这些研究里搜索算法能够高效地使用同强化学习一起训练的线性模型。

新排队问题。很多成功的强化学习应用，比如网络路由 [Littman and Boyan, 1993]、电梯调度 [Crites and Barto, 1996] 和红绿灯控制 [Wiering, 2000]，都是与排队过程很有关系的，这些问题中一些东西需要等待另一些东西。这样的问题有一个优点，即它们通常都能获得直接的反馈。研究对于新排队问题的强化学习方法将会是有趣的，比如火车调度，这个问题中很多不同个体的预期旅行时间被多学习器强化学习技术最小化。很多研究学科会从强化

学习方法中受益，比如远程通信和交通工程，这会使得强化学习解决方案广为接受。

19.2.4 专家对未来发展的看法

最后，作为本书的总结，我们询问了本书的作者们以下几个问题：

- 哪个（哪些）方向将会是强化学习的未来？
- 强化学习中的哪些主题你认为是最重要的，需要更深的理解并且进一步发展？

对于上述问题的答案，我们给出一些专家的最有依据的预测：

Ashvin Shah：因为它们聚焦于通过与环境交互的方式来学习，并且环境中仅具备有限的先验知识和指导，很多强化学习方法深受维度灾难之苦。而包含环境的抽象表示以及对动作的层级表示可以帮助强化学习学习器在一定程度上避免维度灾难。一个很重要的研究主题将是如何自动地建立有用的抽象和层级，这样就可以限制对于先验知识和指导的依赖。这方面研究的灵感可能来自于很多方面，包括动物如何发展和使用这样的表示。产生的研究成果可以生产更多的可行的、灵活的和自动的人工学习器。

624 Lucian Busoniu：基于逼近的算法的进步促进了近年来强化学习领域迅速成熟并且多元化。一个引人注意的问题是令人信服的行业基准仍然稀缺，尤其是在强化学习应用于物理系统的研究领域，这些领域中强化学习的基准问题仍然是简单的或者可以由经典技术解决的问题，比如说爬山问题、倒立摆、自行车平衡问题。物理系统实时监控的一个基本的要求就是安全：学习过程一定不能损害系统或者危害使用者。与控制理论的认知（比如系统稳定性中的认知）一样，安全需求与探索的相互作用预计会成为学习过程的关键问题。

Todd Hester：在强化学习中，能在当下和将来都占有重要地位的研究方向是能够更加实用和高效地在复杂的真实问题中应用强化学习的方向。面向这个目标，我们相信强化学习领域的研究会集中在算法方面，这些算法的目标是同时做到高效地采样和高效地计算。一方面在有限数据甚至是连续状态空间的情况下能够足够高效地采样，另一方面计算要足够高效直到能够实时运行。基于模型的强化学习是一条获得有效的采样的有效途径，并且应该会持续成为焦点。在提升复杂状态空间中基于模型算法的计算效率方面，开发出能够在多台计算机和机器人上使用并行处理器运行的算法是一个很有希望的方向。

Lihong Li：在强化学习中我们需要的是强化学习函数逼近在泛化能力上的数学理论。强化学习领域内重大的进展应是在强化学习探索、模型学习和价值函数学习等方面有理论保证的前提下，系统地使用先验知识。最终我们需要的是脱离策略的健壮的强化学习算法，它们能从一批具有小偏置和方差的数据中学习强化学习。如果算法是在线运行的，无偏估计方法将能从平均意义上揭示出算法中真正的总体奖励量。

Ann Nowe：随着电子设备更多地在自组网中彼此连接，多学习器的强化学习预计会变成一个主流途径来组织这些难以构建出先验模型的系统。一个主要的挑战将是在学习时间方面提升现有算法，有可能通过把单学习器扩展为 MAS 的方法来实现。另外，大多数方法假设了一些特定的环境，比如它们假设所有学习器都有一个公共的兴趣点，或者学习器处在社会困境中。随着环境变得越来越复杂，学习器也许需要引入异构性、未知的交互。因此，需要更多一般方法来解决很多各种不同情境下的问题。

Frans Oliehoek：也许单学习器决策中最大的挑战是需要克服贝尔曼的维度灾难。真实情况中状态的数量是巨大的（通常是无穷的）。因此加深我们对于泛化和函数逼近的理解就变得十分关键，尤其是泛化和逼近与特征选择相结合的方法。我认为放弃状态的概念而聚焦

于子状态（基于特征的）表示是必要的。状态和相应马尔可夫性存在的问题是它们使动态预测问题变得过分简化。这直接导致了维度灾难，并且我相信通过显式学习、表示和挖掘状态变量、特征和相互作用的影响能够缓解这个问题。这个方法也许能够更好地整合来自多学习器规划社区的思想，多学习器规划社区已经探索了“局部状态”（特征的子集）和动作之间的有限影响。

625

Jan Peters：目前强化学习正处于一个十字路口，未来有两个可能的方向：一条路是我们可以加强传统强化学习，并且沿着原来的道路走入难懂深奥的阴霾。另一条路是我们可以走回到基本问题并且使用更加坚实的答案，正如监督学习做的那样，会在应用方面做得更好。我希望强化学习会选择后者。未来强化学习的关键在于把目光集中在强化学习最初的问题上（如 [Peters et al, 2010] 做的那样），并且观察最初的变化如何影响它的对偶。从这个视角来看，一些强化学习方法是完全符合逻辑的，而另一些方法则会出现理论上的缺陷。这些有缺陷的理论也许包含了对于价值函数逼近的代价函数的经典假设，例如 TD 或者贝尔曼误差，这一点在 [Schoknecht, 2002] 中已经指出。第二个部分将会是对于世界的合理假设，人类智能并不是独立于我们生活的这个世界的，机器人的强化学习也不应该这样。在机器人强化学习中，我认为混合离散-连续的机器人强化学习将会出现。

Shimon Whiteson：强化学习未来的研究可能会把重点放在把人类放入这个循环当中。这个方向上已经有人做出了初步的贡献，比如 TAMER 框架，但是人机交互领域中仍有很多方面需要发展，更好地理解人类先验知识能够表达的东西，并且创造出可以从隐式和显式反馈中学习的方法。强化学习领域可以从更丰富的表示和更多实用的探索策略中获益。至今，一些复杂表示比如在演化计算中的间接编码只能用于策略搜索方法，因为使用它们价值函数算法还没有开发出来。除此以外，即使是最有效的探索策略对于很多实际任务来说也是过于危险的。一个重要的目标是开发出在含有大量毁灭性错误威胁的任务中进行安全探索的实用策略。这两个领域是紧密结合的，因为表示应该为引导探索（比如，通过减少它们的不确定性）而设计，并且探索策略应该考虑如何把表示与采样结果相结合。

致谢。我们要感谢 Jurgen Schmidhuber 帮助我们编写了这一章。

参考文献

- Alpaydin, E.: Introduction to Machine learning. The MIT Press (2010)
- Azar, M.G., Munos, R., Ghavamzadeh, M., Kappen, H.J.: Speedy Q-learning. Advances in Neural Information Processing Systems (2011)
- Bakker, B., Schmidhuber, J.: Hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization. In: Proceedings of the 8th Conference on Intelligent Autonomous Systems, IAS-8, pp. 438–445 (2004)
- Baxter, J., Tridgell, A., Weaver, L.: Knightcap: A chess program that learns by combining TD(λ) with minimax search. Tech. rep., Australian National University, Canberra (1997)
- Berliner, H.: Experiences in evaluation with BKG - a program that plays backgammon. In: Proceedings of IJCAI, pp. 428–433 (1977)
- Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-dynamic Programming. Athena Scientific, Belmont (1996)
- Bishop, C.: Pattern Recognition and Machine learning. Springer, Heidelberg (2006)
- Coates, A., Abbeel, P., Ng, A.: Apprenticeship learning for helicopter control. Commun. ACM 52(7), 97–105 (2009)
- Coulom, R.: Efficient Selectivity and Backup Operators in Monte-carlo Tree Search. In: van den Herik, H.J., Ciancarini, P., Donkers, H.H.L.M.(J.) (eds.) CG 2006. LNCS, vol. 4630,

626

- pp. 72–83. Springer, Heidelberg (2007)
- Cramer, N.L.: A representation for the adaptive generation of simple sequential programs. In: Grefenstette, J. (ed.) *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pp. 183–187 (1985)
- Crites, R., Barto, A.: Improving elevator performance using reinforcement learning. In: Touretzky, D., Mozer, M., Hasselmo, M. (eds.) *Advances in Neural Information Processing Systems*, Cambridge, MA, vol. 8, pp. 1017–1023 (1996)
- Di Caro, G., Dorigo, M.: An adaptive multi-agent routing algorithm inspired by ants behavior. In: *Proceedings of PART 1998 - Fifth Annual Australasian Conference on Parallel and Real-Time Systems* (1998)
- Dietterich, T., Wang, X.: Batch value function approximation via support vectors. In: *Advances in Neural Information Processing Systems*, vol. 14, pp. 1491–1498 (2002)
- Dorigo, M., Gambardella, L.M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. *Evolutionary Computation* 1(1), 53–66 (1997)
- Dorigo, M., Maniezzo, V., Coloni, A.: The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B* 26(1), 29–41 (1996)
- Ernst, D., Geurts, P., Wehenkel, L.: Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research* 6, 503–556 (2005)
- Gambardella, L.M., Taillard, E., Dorigo, M.: Ant colonies for the quadratic assignment problem. *Journal of the Operational Research Society* 50, 167–176 (1999)
- van Hasselt, H.: Double Q-learning. In: *Advances in Neural Information Processing Systems*, vol. 23, pp. 2613–2621 (2010)
- Hutter, M.: *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, Berlin (2004)
- Kober, J., Peters, J.: Policy Search for Motor Primitives in Robotics. *Machine Learning* 84(1–2), 171–203 (2011)
- Kolmogorov, A.: Three approaches to the quantitative definition of information. *Problems of Information Transmission* 1, 1–11 (1965)
- Koza, J.R.: Genetic evolution and co-evolution of computer programs. In: Langton, C., Taylor, C., Farmer, J.D., Rasmussen, S. (eds.) *Artificial Life II*, pp. 313–324. Addison Wesley Publishing Company (1992)
- Koza, J.R.: *Genetic Programming II – Automatic Discovery of Reusable Programs*. MIT Press (1994)
- Li, M., Vitányi, P.M.B.: An introduction to Kolmogorov complexity and its applications. In: van Leeuwen, J. (ed.) *Handbook of Theoretical Computer Science*, pp. 188–254. Elsevier Science Publishers B.V (1990)
- Littman, M., Boyan, J.: A distributed reinforcement learning scheme for network routing. In: Alspector, J., Goodman, R., Brown, T. (eds.) *Proceedings of the First International Workshop on Applications of Neural Networks to Telecommunication*, Hillsdale, New Jersey, pp. 45–51 (1993)
- Lowe, D.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 315–333 (2004)
- Maei, H., Szepesvari, C., Bhatnagar, S., Sutton, R.: Toward off-policy learning control with function approximation. In: *Proceedings of the International Conference on Machine Learning*, pp. 719–726 (2010)
- McCallum, R.A.: Instance-based utile distinctions for reinforcement learning with hidden state. In: Prieditis, A., Russell, S. (eds.) *Machine Learning: Proceedings of the Twelfth International Conference*, pp. 387–395. Morgan Kaufmann Publishers, San Francisco (1995)
- McGovern, A., Andrew, G., Barto, E.M.: Scheduling straight-line code using reinforcement learning and rollouts. In: *Proceedings of Neural Information Processing Systems*. MIT Press (1999)
- Mitchell, T.M.: *Machine learning*. McGraw Hill, New York (1996)
- Moore, A.W., Atkeson, C.G.: Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning* 13, 103–130 (1993)

- Moriarty, D.E., Miikkulainen, R.: Efficient reinforcement learning through symbiotic evolution. *Machine Learning* 22, 11–32 (1996)
- Nevmyvaka, Y., Feng, Y., Kearns, M.: Reinforcement learning for optimized trade execution. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 673–680 (2006)
- Nouri, A., Littman, M.: Dimension reduction and its application to model-based exploration in continuous spaces. *Machine Learning* 81(1), 85–98 (2010)
- van Otterlo, M.: Efficient reinforcement learning using relational aggregation. *Proceedings of the Sixth European Workshop on Reinforcement Learning, EWRL-6* (2003)
- Peters, J., Mülling, K., Altun, Y.: Relative entropy policy search. In: Fox, M., Poole, D. (eds.) *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010* (2010)
- Peters, J., Vijayakumar, S., Schaal, S.: Reinforcement Learning for Humanoid Robotics. In: *IEEE-RAS International Conference on Humanoid Robots, Humanoids* (2003)
- Peters, J., Schaal, S.: Reinforcement Learning of Motor Skills with Policy Gradients. *Neural Networks* 21(4), 682–697 (2008), doi:10.1016/j.neunet.2008.02.003
- Poland, J., Hutter, M.: Universal learning of repeated matrix games. In: *Proc. 15th Annual Machine Learning Conf. of Belgium and The Netherlands (Benelearn 2006)*, Ghent, pp. 7–14 (2006)
- Riedmiller, M.: Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) *ECML 2005. LNCS (LNAI)*, vol. 3720, pp. 317–328. Springer, Heidelberg (2005)
- Riedmiller, S., Riedmiller, M.: A neural reinforcement learning approach to learn local dispatching policies in production scheduling. In: *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI 1999)* (1999)
- Ring, M.: Continual learning in reinforcement environments. PhD thesis, University of Texas, Austin, Texas (1994)
- Safustowicz, R.P., Schmidhuber, J.H.: Probabilistic incremental program evolution. *Evolutionary Computation* 5(2), 123–141 (1997)
- Schmidhuber, J.: The Speed Prior: A New Simplicity Measure Yielding Near-Optimal Computable Predictions. In: Kivinen, J., Sloan, R.H. (eds.) *COLT 2002. LNCS (LNAI)*, vol. 2375, pp. 216–228. Springer, Heidelberg (2002)
- Schmidhuber, J.: Optimal ordered problem solver. *Machine Learning* 54, 211–254 (2004)
- Schmidhuber, J.: Ultimate cognition *à la* Gödel. *Cognitive Computation* 1(2), 177–193 (2009)
- Schmidhuber, J.: Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development* 2(3), 230–247 (2010)
- Schmidhuber, J., Zhao, J., Schraudolph, N.: Reinforcement learning with self-modifying policies. In: Thrun, S., Pratt, L. (eds.) *Learning to Learn*, pp. 293–309. Kluwer (1997a)
- Schmidhuber, J., Zhao, J., Schraudolph, N.N.: Reinforcement learning with self-modifying policies. In: Thrun, S., Pratt, L. (eds.) *Learning to Learn*. Kluwer (1997b)
- Schmidhuber, J.H.: Temporal-difference-driven learning in recurrent networks. In: Eckmiller, R., Hartmann, G., Hauske, G. (eds.) *Parallel Processing in Neural Systems and Computers*, pp. 209–212. North-Holland (1990)
- Schmidhuber, J.H.: Curious model-building control systems. In: *Proceedings of the International Joint Conference on Neural Networks*, vol. 2, pp. 1458–1463. IEEE, Singapore (1991a)
- Schmidhuber, J.H.: A possibility for implementing curiosity and boredom in model-building neural controllers. In: Meyer, J.A., Wilson, S.W. (eds.) *Proceedings of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pp. 222–227. MIT Press/Bradford Books (1991b)
- Schmidhuber, J.H.: A general method for incremental self-improvement and multi-agent learning in unrestricted environments. In: Yao, X. (ed.) *Evolutionary Computation: Theory and Applications*. Scientific Publ. Co., Singapore (1996)
- Schmidhuber, J.H., Zhao, J., Wiering, M.A.: Shifting inductive bias with success-story algo-

- rithm, adaptive Levin search, and incremental self-improvement. *Machine Learning* 28, 105–130 (1997c)
- Schoknecht, R.: Optimality of reinforcement learning algorithms with linear function approximation. In: Becker, S., Thrun, S., Obermayer, K. (eds.) *Advances in Neural Information Processing Systems, NIPS 2002*, pp. 1555–1562 (2002)
- van Seijen, H., Whiteson, S., van Hasselt, H., Wiering, M.: Exploiting best-match equations for efficient reinforcement learning. *Journal of Machine Learning Research* 12, 2045–2094 (2011)
- Simpkins, C., Bhat, S., Isbell Jr., C., Mateas, M.: Towards adaptive programming: integrating reinforcement learning into a programming language. *SIGPLAN Not.* 43, 603–614 (2008)
- Singh, S., Litman, D., Kearns, M., Walker, M.: Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. *Journal of Artificial Intelligence Research* 16, 105–133 (2002)
- Smart, W., Kaelbling, L.: Effective reinforcement learning for mobile robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3404–3410 (2002)
- Solomonoff, R.: A formal theory of inductive inference. Part I. *Information and Control* 7, 1–22 (1964)
- Solomonoff, R.: Complexity-based induction systems. *IEEE Transactions on Information Theory* IT-24(5), 422–432 (1978)
- Sutton, R.S.: Learning to predict by the methods of temporal differences. *Machine Learning* 3, 9–44 (1988)
- Sutton, R.S.: Integrated architectures for learning, planning and reacting based on dynamic programming. In: *Machine Learning: Proceedings of the Seventh International Workshop* (1990)
- Sutton, R.S., Precup, D., Singh, S.P.: Between MDPs and semi-MDPs: Learning, planning, learning and sequential decision making. Tech. Rep. COINS 89-95, University of Massachusetts, Amherst (1998)
- Tanner, B., White, A.: RL-Glue: Language-independent software for reinforcement-learning experiments. *Journal of Machine Learning Research* 10, 2133–2136 (2009)
- Tesauro, G.: Temporal difference learning and TD-Gammon. *Communications of the ACM* 38, 58–68 (1995)
- Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer, Heidelberg (1995)
- Veness, J., Silver, D., Uther, W., Blair, A.: Bootstrapping from game tree search. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) *Advances in Neural Information Processing Systems*, vol. 22, pp. 1937–1945 (2009)
- Veness, J., Ng, K., Hutter, M., Uther, W., Silver, D.: A Monte-carlo AIXI approximation. *Journal of Artificial Intelligence Research* (2011)
- Watkins, C.J.C.H.: Learning from delayed rewards. PhD thesis, King's College, Cambridge, England (1989)
- Watkins, C.J.C.H., Dayan, P.: Q-learning. *Machine Learning* 8, 279–292 (1992)
- Westra, J.: Organizing adaptation using agents in serious games. PhD thesis, Utrecht University (2011)
- Wiering, M.: Self-play and using an expert to learn to play backgammon with temporal difference learning. *Journal of Intelligent Learning Systems and Applications* 2(2), 57–68 (2010)
- Wiering, M., van Hasselt, H.: Ensemble algorithms in reinforcement learning. *IEEE Transactions, SMC Part B, Special Issue on Adaptive Dynamic Programming and Reinforcement Learning in Feedback Control* (2008)
- Wiering, M., van Hasselt, H.: The QV family compared to other reinforcement learning algorithms. In: *Proceedings of the IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL 2009)*, pp. 101–108 (2009)
- Wiering, M.A.: Multi-agent reinforcement learning for traffic light control. In: Langley, P. (ed.) *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 1151–1158 (2000)
- Wiering, M.A., Schmidhuber, J.H.: Solving POMDPs with Levin search and EIRA. In: Saitta, L. (ed.) *Machine Learning: Proceedings of the Thirteenth International Conference*, pp. 534–542. Morgan Kaufmann Publishers, San Francisco (1996)

缩 写 词

AC (Actor-Critic)	演员 – 评论家
AO (Action-Outcome)	动作 – 结果
BAC (Bayesian Actor-Critic)	贝叶斯演员 – 评论家
BEETLE (Bayesian Exploration-Exploitation Tradeoff in Learning)	学习中的贝叶斯探索 – 运用平衡
BG (Basal Ganglia)	基底神经节
BQ (Bayesian Quadrature)	贝叶斯正交
BQL (Bayesian Q-learning)	贝叶斯 Q 学习
BPG (Bayesian Policy Gradient)	贝叶斯策略梯度
BRM (Bellman Residual Minimization (generic; BRM-Q for Q-functions; BRM-V for V-functions))	贝尔曼残差最小化 (通用; Q 函数为 BRM-Q; V 函数为 BRM-V)
CMA-ES (Covariance Matrix Adaptation Evolution Strategy)	协方差矩阵自适应演进策略
CPPN (Compositional Pattern Producing Network)	组合模式生成网络
CoSyNE (Cooperative Synapse Coevolution)	协同演化
CR (Conditioned Response)	条件反射
CS (Conditioned Stimulus)	条件刺激
DA (Dopamine)	多巴胺
DBN (Dynamic Bayesian Network)	动态贝叶斯网络
DEC-MDP (Decentralized Markov Decision Process)	去中心化的马尔可夫决策过程
DFQ (Deep Fitted Q iteration)	深度拟合的 Q 迭代
DP (Dirichlet process)	狄利克雷过程
DP (Dynamic Programming)	动态规划
DTR (Decision-Theoretic Regression)	决策理论回归
EDA (Estimation of Distribution Algorithm)	分布估计算法
ESP (Enforced SubPopulations)	强制子种群
FODTR (First-Order (Logical) Decision-Theoretic Regression)	一阶决策 – 理论回归
FQI (Fitted Q Iteration)	拟合的 Q 迭代
GP (Gaussian Process)	高斯过程
GPI (Generalized Policy Iteration)	广义策略迭代
GPTD (Gaussian Process Temporal Difference)	高斯过程时序差分
HBM (Hierarchical Bayesian model)	层次式贝叶斯模型
HRL (Hierarchical Reinforcement Learning)	层次式强化学习
ILP (Inductive Logic Programming)	
KADP (Kernel-based Approximate Dynamic Programming)	基于核的近似动态规划
KR (Knowledge Representation)	知识表示
KWIK (Knows What It Knows)	知其然
LCS (Learning Classifier System)	学习分类器系统

LSPE (Least-Squares Policy Evaluation (generic; ; LSPE-Q for Q-functions; LSPE-V for V-functions))	最小二乘策略评估 (通用; Q 函数为 LSPE-Q; V 函数为 LSPE-V)
LSPI (Least-Squares Policy Iteration)	最小二乘策略迭代
LSTDQ (Least-Squares Temporal Difference Q-Learning)	最小二乘时序差分 Q 算法
LSTD (Least-Squares Temporal Difference (generic; LSTD-Q for Q-functions; LSTD-V for V-functions))	最小二乘时序差分算法 (通用; Q 函数为 LSTD-Q; V 函数为 LSTD-V)
MB (Mistake Bound)	误差边界
MC (Monte-Carlo)	蒙特卡罗
MCTS (Monte Carlo Tree Search)	蒙特卡罗搜索树
MDP (Markov Decision Process)	马尔可夫决策过程
ML (Machine Learning)	机器学习
MTL (Multi-Task Learning)	多任务学习
MTRL (Multi-Task Reinforcement Learning)	多任务强化学习
NEAT (Neuro Evolution of Augmenting Topologies)	神经演化强化拓扑
NFQ (Neural Fitted Q iteration)	神经拟合的 Q 迭代
PAC (Probably Approximately Correct)	概率近似正确
PAC-MDP (Probably Approximately Correct in Markov Decision Process)	概率近似正确的马尔可夫决策过程
PMBGA (Probabilistic Model-Building Genetic Algorithm)	概率建模遗传算法
PI (Policy Iteration)	策略迭代
PIAGeT (Policy Iteration using Abstraction and Generalization Techniques)	基于抽象化和一般化的策略迭代
POMDP (Partially Observable Markov Decision Process)	部分可观察的马尔可夫决策过程
RL (Reinforcement Learning)	强化学习
RMDP (Relational Markov Decision Process)	关系型马尔可夫决策过程
SANE (Symbiotic Adaptive NeuroEvolution)	共生适应性神经演化
sGA (Structured Genetic Algorithm)	结构遗传算法
SMDP (Semi-Markov Decision Process)	半马尔可夫决策过程
SR (Stimulus-Response)	刺激反应
SRL (Statistical Relational Learning)	统计关系学习
TD (Temporal Difference)	时序差分
TWEANN (Topology- and Weight-Evolving Artificial Neural Network)	基于拓扑和基于权重的人工神经网络
UR (Unconditioned Response)	非条件 (无条件) 反应
US (Unconditioned Stimulus)	非条件刺激
VI (Value Iteration)	值迭代
VPI (Value of Perfect Information)	完美信息价值

索引

索引中的页码为英文原书页码,与书中页边标注的页码一致。

E^3 , 121

α , 30

α -functions (α -函数), 374

γ , 14

A

$\alpha\beta$ -pruning ($\alpha\beta$ -剪枝), 551

absorbing state (吸收状态), 13, 121

abstract action (抽象动作)

sequential (序列的), 316

abstract actions (抽象动作), 297

abstraction (抽象)

in Poker (扑克游戏), 564

action (动作), 10

applicable (可应用的), 10

precondition (前提条件), 11

action language (动作语言), 256

action-selection networks (动作-选择网络), 329

actor critic (演员评论家), 212, 230, 234

ADHDP, 236

Cacla, 236, 238

natural actor critic (自然的演员评论家), 235, 238

actor only (仅限于演员), 230

actor-critic algorithm (演员-评论家算法), 32

Actor-critic methods (演员-评论家方法), 369

agent (学习器, 智能体), 5

agnostic (不可知论者), 189

alternating maximization (交替最大化), 481

approximate linear programming (近似线性规划)

in Tetris (俄罗斯方块), 557

approximate policy iteration (近似策略迭代), 78

optimistic (乐观的), 89

arms race (军备竞赛), 342

artificial general intelligence (通用人工智能), 293

asymptotic improvement (渐近的改进), 153

asynchronous updating (异步更新) 25

automatic feature construction (自动特征创建), 563

average loss (平均损失), 182

B

Backgammon (西洋双陆棋), 542

Neurogammon, 543

TD-Gammon, 543

backpropagation (反向传播), 543, 546

Tangent-Prop (Tangent-Prop 算法), 546

backup operator (备份操作), 21, 24

Baldwin effect (鲍德温效应), 335

ball-in-a-cup (ball-in-a-cup 游戏), 599

basal ganglia (基底核), 521

basis function (基函数), 81

batch learning (批处理学习), 45

algorithms (算法), 52

applications (应用), 69

problem (问题), 46

Bayesian actor-critic (贝叶斯演员-评论家), 369

Bayesian DP (贝叶斯动态规划), 127

Bayesian game (贝叶斯游戏), 486

Bayesian Multi-Task Reinforcement Learning (贝叶斯多任务强化学习), 377

Bayesian policy gradient (贝叶斯策略梯度), 368

Bayesian Q-learning (贝叶斯 Q 学习), 361

Bayesian quadrature (贝叶斯积分), 366

Bayesian reinforcement learning (贝叶斯强化学习), 359

BEB, 128

BEETLE, 127, 376

behavioral cloning (行为克隆), 36

belief (信念), 392

Belief monitoring (信念监测), 373

belief state MDP (信念状态 MDP), 127

belief update (信念更新), 392
 Bellman backup (贝尔曼备份), 374
 Bellman equation (贝尔曼等式), 16, 77, 373
 projected (投影的, 见 projected Bellman equation)
 Bellman optimality equation (贝尔曼最优化等式), 16
 Bellman residual minimization (贝尔曼残差最小化),
 80, 84
 best response policy (最佳反映策略), 568
 Bgblitz, 543
 bias-variance tradeoffs (偏置 - 方差平衡), 381
 binary action search (二值动作搜寻), 132
 Blocks World (积木游戏), 258
 bootstrapping (bootstrapping 算法), 30
 BOSS, 127
 bounded policy iteration (边界策略迭代), 496
 branching factor (分支因子), 542

C

Cacla, 236, 238
 cart pole (车杆)
 double-pole cart pole (双极车杆), 238
 cart-pole balancing (车杆平衡), 133
 case-based learning (基于实例的学习), 570
 in real-time strategy games (在实时策略游戏中),
 560
 cellular encodings (细胞编码), 343
 Checkers (跳棋), 548, 564
 Chess (国际象棋), 545
 knight fork (骑士叉), 549
 Morph (变身), 564
 NeuroChess (神经元国际象棋), 546
 tutoring (辅导), 566
 UCT, 555
 classical conditioning (经典条件反射), 508
 CMA-ES, 221, 238, 330
 in Tetris (在俄罗斯方块中), 557
 cognitive architecture (认知构架), 257
 communication (通信), 497
 competing conventions problem (近亲繁殖的问题),
 332
 competitions (挑战赛)
 AIIDE Starcraft Competition (AIIDE 星际争霸
 挑战赛), 561

 Ms. Pac-Man Competition (Ms. Pac-Man 挑战
 赛), 562
 competitive fitness sharing (竞争适应度共享), 342
 complexity of dynamic programming (动态规划的
 复杂性), 24
 concept drift (概念漂移), 9
 confidence interval (置信区间), 552
 counterfactual regret (与事实相反的遗憾), 567
 continual complexification (连续性复杂化), 342
 continuous action space (连续动作空间), 208
 continuous state space (连续状态空间), 208
 contraction mapping (收缩映射), 24
 CoSyNE, 341
 CPPNs, 344
 credit assignment (信用分配), 7
 cross-entropy method (交叉熵方法)
 in Tetris (在俄罗斯方块), 557
 cross-entropy optimization (交叉熵优化), 221, 493
 curse of dimensionality (维度诅咒), 582

D

Darwinian evolution (达尔文进化论), 334
 DBN- E^3 , 124
 Dec-POMDP, 471
 brute force search (暴力搜索), 480
 decision rule (决策规测), 477
 dynamic programming (动态规划), 489
 factored (因素分解的), 494
 history (历史), 475
 infinite horizon (无限域), 495
 model (模型), 473
 optimal Q-value function (最优 Q 值函数), 484
 policy (策略), 476
 TOI, 494
 transition independent (过渡独立), 494
 Dec-POMDP-Com, 497
 decentralized POMDP (去中心化的 POMDP), 471
 decision-theoretic planning (决策理论规划), 18
 decision-theoretic regression (决策理论回归), 262
 Deep Blue (深蓝), 545
 Deep Fitted Q Iteration (深度拟合 Q 迭代), 67
 delayed feedback (延迟的反馈), 7
 density estimation (密度估计), 330

- DFQ, 67
- difference functions (差分函数), 340
- difficulty scaling (难度调整), 569
- direct policy search (直接策略搜索), 212, 234, 556
- dynamic scripting (动态脚本), 559
- partial observability (部分可观察性), 567
- Dirichlet distribution (Dirichlet 分布), 372
- Dirichlet mixture (Dirichlet 混合), 379
- Dirichlet process (Dirichlet 过程), 380
- discount factor (阻尼系数), 14
- diversity of gameplay (多样性的游戏), 569
- domain knowledge (领域知识), 360, 553
- in games (在游戏中), 561
- dopamine (多巴胺对), 518
- DRE, 132
- DYNA, 38
- Dyna, 118
- Dyna-2, 551
- Dyna-2, 119
- Dynamic Bayesian Network (动态贝叶斯网络), 122
- Dynamic Programming (动态规划), 373
- dynamic programming (动态规划), 19
- Dec-POMDP, 489
- memory-bounded (内存受限制的), 492
- point-based (基于点的), 491
- dynamic scripting (动态脚本), 570
- difficulty scaling (难度脚本), 570
- in real-time strategy games (即时战略游戏), 559
- E**
- EDI-CR, 494
- eligibility trace (资格迹), 37
- eligibility traces (资格迹), 37
- performance in Backgammon (西洋双陆棋的表现), 545
- Elo rating (ELO 等级), 545
- of patterns (模式), 551
- envelope (信封), 26
- environment (环境), 5
- episodic task (阶段性任务), 12
- ESP, 341
- estimation of distribution algorithms (分布估计算法), 330
- evaluative feedback (评估型反馈), 8
- event-driven Dec-MDP (事件驱动的 Dec-MDP), 494
- evolutionary algorithms (进化算法), 221
- evolutionary computation (进化计算), 325
- coevolution (进化), 339
- neuroevolution (神经进化), 327
- on-line (在线的), 346
- steady-state (稳定状态的), 338
- evolutionary function approximation (进化函数逼近), 335
- evolutionary game theory (进化的博弈理论), 550
- evolutionary learning (进化学习)
- in Starcraft (在 Starcraft 游戏中), 560
- in Tetris (在 Tetris 游戏中), 557
- in Wargus (在 Wargus 游戏中), 560
- evolutionary strategies (进化策略), 221, 234
- CMA-ES, 221, 238
- natural evolutionary strategies (自然进化策略), 221, 234
- EvolutionChamber, 560
- experiment replay (经验重放), 50
- experiment (经验)
- double-pole cart pole (双极车杆), 238
- explicit fitness sharing (显式适应度共享), 334
- exploration (探索), 28, 34, 88, 126
- automatic (自动的), 544
- Boltzmann, 29
- Boltzmann exploration (Boltzmann 探索), 232
- ϵ -greedy (ϵ 贪婪), 565
- Gaussian exploration (高斯探索), 232
- in games (在游戏中), 564
- softmax, 29
- exploration gain (探索收获), 377
- exploration overhead (探索延迟), 49
- exploration-exploitation tradeoff (探索-利用平衡), 8
- exploration/exploitation tradeoff (探索/利用平衡), 360, 373
- F**
- factored Dec-POMDP (分解因子 Dec-POMDP), 494
- factored state (分解因子状态), 122
- factored-r-max (分解因子 r-max), 124

- family of relational MDPs (关系型 MDP 家族), 260
 - feature transfer (特征转移), 157, 161, 162
 - finite horizon task (有限域任务), 12
 - Finite Sample Analysis (有限样本分析), 380
 - first-order (logical) basis functions (一阶(逻辑)基函数), 266
 - first-order decision-theoretic regression (一阶决策理论回归), 264
 - Fisher kernels (Fisher 核), 368
 - fitness function (拟合函数), 325
 - Fitted Q Iteration (拟合 Q 循环), 55
 - Fitted-R-Max (拟合 R 最大化), 132
 - fitting (拟合), 50
 - fog of war (战争迷雾), 566
 - forward-sweep policy computation (前掠策略计算), 485
 - FQI, 55
 - fun gameplay (有趣的游戏), 568
 - balanced strategies (平衡策略), 571
 - function approximation (函数拟合), 34, 208, 212, 592
 - discretization (离散化), 214
 - fuzzy sets (模糊集), 217
 - in Tetris (在 Tetris 游戏中), 556
 - linear (线性的), 213
 - non-linear (非线性的), 217
 - tile coding (复制编码), 214
- ## G
- game balance (游戏平衡), 571
 - games (游戏), 539
 - Atari 2600 games (Atari 2600 游戏), 562
 - Backgammon (双陆棋), 542
 - Baldur's Gate (博德之门(游戏)), 569
 - Black & White, 570
 - Bridge, 567
 - Checkers, 564
 - Chess (象棋), 545
 - Civilization (文明), 570
 - Creatures, 570
 - Diplomacy (外交), 564
 - general gameplay (一般性游戏), 564
 - Go (围棋), 550
 - Hearts, 564
 - Knock'em, 569
 - Last Night on Earth (世界末夜), 571
 - Magic: The Gathering (万智牌游戏), 571
 - Ms. Pac-Man (派克曼电子游戏软件), 564
 - NERO, 570
 - Neverwinter Nights (绝冬城之夜(游戏)), 570, 571
 - Othello (奥赛罗), 564
 - Poker (扑克), 564, 568
 - Pong, 569
 - real-time strategy (实时策略游戏), 558
 - Suicide chess (自杀象棋), 550
 - SZ-Tetris (俄罗斯方块), 556
 - teaching games (教育游戏), 570
 - Tetris, 555
 - Gaussian process (高斯过程), 363
 - Gaussian Process Temporal Difference (时序差分高斯过程), 363
 - generalization (一般化), 114
 - generalized multi-agent A* (一般化的多代理 A*), 488
 - generalized policy iteration (一般化策略迭代), 18
 - generative and developmental systems (生成和发展系统), 327, 343
 - generative model (生成模型), 113
 - genetic algorithms (基因算法), 325, 493
 - GMAA*, 488
 - GNUbg, 543
 - GNUchess, 547
 - Go (围棋), 550
 - nakade, 555
 - tutoring (辅导), 566
 - goal specification (目标规范), 585
 - goal state (目标状态), 12
 - gradient descent (梯度下降), 219
 - gradient free optimization (无梯度优化), 220
 - gradient temporal-difference learning (梯度时序差分学习), 226
 - growing batch learning (增长批处理学习)
 - problem (问题), 48
 - guidance (指导), 36
 - heuristics (启发式), 26
 - equivalent experience (相关经验), 553
 - RAVE, 553

virtual wins/losses (虚拟输赢), 553

HEXQ

exit (退出), 315

hierarchical reinforcement learning (HRL) (层次式强化学习 (HRL)), 295

HAMQ, 307

HEXQ, 315

MAXQ, 309

options (选择), 306

host/parasite model (宿主/寄生模型), 342

hyperparameters (超参数), 373

imitation (模仿), 594

incremental policy generation (增量策略生成), 491

indefinite horizon task (无限域任务), 12

independent Q-learners (独立 Q 学习器), 496

indirect encodings (非直接编码), 343

initial state distribution (起始状态分布), 12

innovation numbers (创新的数字), 333

instance transfer (实例迁移), 150

instructive feedback (指导性反馈), 8

intensional dynamic programming (内涵动态规划), 262

international planning competition (国际规划竞赛), 267

inverse reinforcement learning (逆向强化学习), 585

iterated elimination of dominated policies (迭代消除占主导地位的策略), 491

J

JESP, 481

joint action (联合行动), 445

joint belief (联合置信), 482, 498

jumpstart improvement (启动增强), 153

KADP, 52

kernel-based approximate dynamic programming (基于核的近似动态规划), 52

KnightCap, 547

known state-action MDP (已知状态-动作 MDP), 184, 187, 192, 198

empirical known state-action MDP (经验已知状态-动作 MDP), 184

KWIK, 121, 189

Knows What It Knows (知之为知之), 189

KWIK-learnable (可学习的 KWIK), 190

KWIK-learnable MDPs (可学习 KWIK 的 MDP), 192

L

λ -policy iteration (λ 策略迭代)

in Tetris (俄罗斯方块游戏中), 557

L-systems (L 系统), 343

Lamarckian evolution (Lamarckian 进化), 334

Law of Effect (效应定理), 334

learning classifier systems (学习分类器系统), 335, 336

anticipatory (期待的), 346

Michigan-style (密歇根州风格), 337

Pittsburgh-style (匹兹堡风格), 337

learning rate (学习率), 30

learning speed improvement (学习率提升), 151

least-squares policy evaluation (最小二乘策略评估), 84

least-squares policy iteration (最小二乘策略迭代), 57, 86

in Tetris (俄罗斯方块), 557

online (在线), 90

least-squares temporal difference (最小二乘时序差分), 84

lifted first-order reasoning (提升一阶推理), 266

logical generalization (逻辑一般化), 257

logical language (逻辑语言), 258

logical matching (逻辑匹配), 263

logical regression (逻辑回归), 263

lookahead search (前瞻搜索)

multi-step (多步骤), 547

LSE-R-Max, 124

LSPI, 57, 132

M

MAA*, 488

Markov (马尔可夫), 11

Markov decision process (马尔可夫决策过程), 10, 12

Markov game (马尔可夫游戏), 11

- definition (定义), 455
 - Markov property (马尔可夫特性), 456
 - maximum likelihood estimation (极大似然估计), 368
 - MBBE, 128
 - memory-bounded dynamic programming (内存受限动态规划), 492
 - Met-R-Max, 124
 - micromanagement (微管理), 570
 - minimax (极小化极大), 342
 - minimax search (极小化极大搜索), 547, 555
 - minimax-optimal strategy (极小极大优化策略), 568
 - missing information (丢失的信息)
 - in games (游戏), 566
 - model (模型), 12
 - learning of (学习), 33, 38
 - model approximation (模型逼近), 211
 - model errors (模型错误), 584
 - model-based algorithm (基于模型的算法), 17, 19
 - model-based control (基于模型的控制), 516
 - Model-Based Reinforcement Learning (基于模型的强化学习), 372
 - model-free algorithm (无模型算法), 17, 27
 - model-free control (无模型的控制), 516
 - Model-free RL (无模型的强化学习), 316
 - modified policy iteration (修改策略迭代), 25
 - monomial basis (单项式基), 376
 - Monte Carlo sampling (蒙特卡罗抽样), 33
 - Monte Carlo Tree Search (蒙特卡罗树搜索), 115
 - Monte-Carlo (蒙特卡罗), 366
 - hidden information (隐藏信息), 567
 - value estimation (价值估计), 552
 - Monte-Carlo tree search (蒙特卡罗树搜索), 552
 - in Chess (国际象棋游戏中), 549, 555
 - in Go (围棋游戏中), 553
 - Morph (变身), 549, 564
 - multi-agent A* (多学习器 A*), 488
 - multi-agent belief (多学习器置信), 490, 491
 - multi-agent MDP (多学习器 MDP), 494
 - multi-agent reinforcement learning (多学习器强化学习), 496
 - Multi-task learning (多任务学习), 377
 - myopic policy (短视的策略), 377
- N
- Nash equilibrium (纳什均衡), 481, 49
 - Nash-equilibrium (纳什均衡), 568
 - natural actor critic (自然演员-评论家), 235, 238
 - natural evolutionary strategies (自然进化策略), 221, 234
 - natural gradient (自然梯度), 220
 - natural policy gradient (自然策略梯度), 233
 - ND-POMDP, 495
 - NEAT, 333
 - Neural Fitted Q Iteration (神经拟合 Q 迭代), 61
 - neural network (神经网络), 543, 546
 - as automatic feature construction (自动特征构造), 563
 - as predictive model (预测模型), 546
 - for pattern evaluation (模式评价), 551
 - neural networks (神经网络), 327
 - feed-forward (前项反馈), 329
 - recurrent (循环的), 329
 - NeuroChess, 546
 - Neurogammon, 543
 - neurons (神经元), 340
 - Neverwinter Nights (无冬之夜), 570, 571
 - NFQ, 61
 - niching (小生境), 332
- O
- object (对象), 253, 257
 - object-oriented RL (面向对象的强化学习), 137
 - objective functions (目标函数), 224
 - observation model (观察模型), 390
 - off-policy learning (策略学习), 32
 - offline learning (离线学习), 7
 - on-policy learning (策略学习), 31
 - Online algorithms (在线算法), 376
 - online learning (在线学习), 7
 - operant conditioning (操作条件), 513
 - opponent modelling (反向建模), 567
 - optimal (最优的)
 - hierarchical greedy (层次贪婪), 304
 - hierarchically optimal (层次优化), 304
 - recursively optimal (递归优化), 304

optimism in the face of uncertainty (面对不确定性的乐观), 181, 186, 197
 optimistic value initialization (最佳价值初始化), 29
 option transfer (期权转让), 156, 161
 ORTS, 559
 ORTS Competition (ORTS 挑战赛), 561
 Othello (奥赛罗), 564

P

PAC-Bayesian approach (PAC-贝叶斯方法), 382
 PAC-MDP, 179, 382
 parallel tasks (并行任务), 558
 parameter transfer (参数迁移), 151
 parameterized action (参数化动作), 257
 Pareto optimal (Pareto 最优化), 494
 Pareto optimality (Pareto 最优性), 342
 parti-game (部分游戏), 131
 partial observability (部分可观察性), 388
 partial visibility (部分的可见性), 566
 partially observable Markov decision process (部分可观察马尔可夫决策过程), 11, 372, 390
 partially observable stochastic game (部分可观察随机博弈), 493
 patterns (模式), 551, 554
 local (本地的), 549, 551
 move patterns (移动模式), 549
 team of (队伍), 551
 perceptual aliasing (感性混合), 388
 Perseus (珀耳修斯), 376
 PIAGeT, 254
 PILCO, 131
 planning (规划), 6, 18, 115
 player ranking (玩家排名)
 Elo rating (ELO 等级), 545
 rollout analysis (部署分析), 543
 tournament (赛), 543
 playtesting (游戏测试), 571
 point-based dynamic programming Dec-POMDP (基于点的动态规划 Dec-POMDP), 491
 point-based value iteration (基于点的值迭代), 376
 Poker (扑克), 564, 568
 state aggregation (状态聚集), 564
 policy (策略), 13

ϵ -greedy (ϵ 贪婪), 29
 application of (应用于), 13
 deterministic (定性的), 13
 greedy (贪婪的), 17, 21
 optimal (优化的), 13, 16
 stochastic (随机的), 13
 policy approximation (策略逼近), 212, 229
 policy evaluation (策略评价), 18, 20, 77
 least-squares (最小二乘法, 见 least-squares policy evaluation)
 projected (映射的, 见 projected policy evaluation Policy gradient), 365
 policy gradient (策略梯度), 230, 366
 natural policy gradient (自然策略梯度), 233
 policy gradient theorem (策略梯度理论), 370
 policy improvement (策略提升), 18, 21, 78
 policy iteration (策略迭代), 20, 77
 approximate (拟合, 见 approximate policy iteration)
 least-squares (最小二乘, 见 least-squares policy iteration)
 policy search (策略寻找), 588
 POMDP, 390
 Bellman equation (贝尔曼方程), 394
 learning internal memory (学习内部存储), 405
 model-based techniques (基于模型的技巧), 395
 model-free techniques (无模型的技巧), 404
 point-based methods (基于点的方法), 401
 value function (价值函数), 394
 value iteration (值迭代), 398
 POMDPs, 390
 deep-memory (深度记忆), 341
 Pong, 569
 POSG, 493
 posterior Gaussian process (后置高斯过程), 364
 preference function (偏好函数), 556, 557
 Prior Knowledge (先验知识), 379
 prior knowledge (前置知识), 379
 prioritized sweeping (优先清扫), 33, 38
 probabilistic inference for planning (计划的概率推理), 280
 probabilistic model-building genetic algorithms (建立概率模型的遗传算法), 330
 probabilistic programming languages (概率优化语言), 281

probabilistic STRIPS operator (概率 STRIPS 操作子), 260
 probimax search (probimax 搜查), 568
 progressive unpruning/widening (渐近式非剪枝/扩大), 553
 projected Bellman equation (映射贝尔曼函数), 80
 projected policy evaluation (映射策略评估), 80
 projections (映射), 224
 projected Bellman equation (映射贝尔曼函数), 225

Q

Q-function (Q 函数), 17
 Q-learning (Q 学习), 31, 227
 difficulty scaling (按比例增减难度), 569
 in Chess (在国际象棋中), 546
 QBG, 487
 QMDP, 487
 QPOMDP, 487
 quiescence search (静止搜索), 547

R

R-IAC, 130
 R-Max, 121
 randomness (随机), 544, 556
 RAVE, 553
 real-time architecture (实时架构), 119
 real-time dynamic programming (实时动态规划), 27
 real-time strategy game (实时策略游戏), 558
 difficulty scaling (难度变化), 569
 knight's rush (骑士的匆忙), 559
 seven-roach rush (七蟑螂的匆忙), 560
 real-world interactions (真实世界交互), 584
 real-world samples (真实世界样本), 583
 realizable (可实现的), 189
 regret (遗憾), 180
 reinforcement learning (强化学习), 27
 in commercial games (在商业游戏中), 570
 relation (关系), 253, 257
 relational (关系的)
 decision list (决策列表), 261
 decision tree (决策树), 260

 interpretation (解释), 259
 learning world models (学习世界模型), 277
 Markov decision process (马尔可夫决策过程), 259
 policy (策略), 261
 policy search (策略搜索), 274
 POMDP, 282
 reward function (奖励函数), 260
 value function (价值函数), 260
 relational reinforcement learning (关系型强化学习), 254, 268
 relational representation (关系型代表), 257
 in games (在游戏中), 563
 representation (代表), 9, 589
 automatic feature construction (自动特征构建), 563
 combination features (特征融合), 563
 expert-features (专家特征), 563
 hard to learn (难于学习), 564
 in Backgammon (在 Backgammon 游戏中), 544
 in Chess, 546
 relational (关系的), 563
 representation transfer (代表转换), 150
 Rescorla-Wagner model (Rescorla-Wagner 模型), 511
 reward (奖励)
 local reward (本地奖励), 551
 reward function (奖励函数), 11, 12
 reward model (奖励模型)
 average reward (平均奖励), 14
 finite horizon (有限域), 14
 infinite horizon (无限域), 14
 rich evaluative feedback (丰富的评价反馈), 562
 risk-sensitive optimization (风险敏感度优化), 562
 RL-DT, 125
 robot (机器人), 579
 robust control (鲁棒控制), 381
 roll-out (部署), 115
 rollout (部署), 115
 rollout analysis (部署分析), 543
 rollout policy (部署策略), 553
 balanced (平衡的), 554
 RSPSA, 565

S

sample complexity of exploration (探测样本的复杂性), 178

sample efficiency (采样效率), 120

SANE, 340

SARSA, 31

Sarsa, 227

 in Neverwinter Nights (无冬之夜游戏), 571

search (搜索), 26

self-play (自我对弈), 543, 547, 548, 565

semantics of communication (通信的语义学), 497

semi Markov Decision Problem (SMDP) (半MDP), 297

sequence form (序列形式), 491, 493

sequential decision making (序贯决策), 5

sGA, 331

shaping (整形), 36

simulation (模拟), 596

simulation balancing (模拟平衡), 554

SLF-R-Max, 124

Snowie, 543

sparse sampling (稀疏取样), 116, 377

speciation, 332

SPITI, 125

Starcraft (Starcraft 游戏), 558

state (状态), 10

 value (价值), 15

state abstraction (状态抽象), 301

 eliminating irrelevant variables (消除无关变量), 301

 funnelling (漏斗), 302

state aggregation (聚集状态), 563

state representation (状态表达)

 atomic (原子), 255

 deictic (指示), 255

 propositional (命题), 255

 relational (关系的), 563

state-action value function (状态-动作价值函数), 17

statistical relational learning (统计关系学习), 256

stochastic bisimulation homogeneity (随机双模拟的同质化), 302

Stratagus (Stratagus 游戏), 558

structural credit assignment (结构信用分配), 8

structure learning (结构学习), 123

structured Bellman backup (结构贝尔曼备份), 262

sub-tree policy (子树策略), 478

substitution (替代), 259

subsumption (包含), 259

Suicide chess (自杀国际象棋), 550

synchronous updating (同步更新), 25

SZ-Tetris, 556, 558

T

tabular model (表格模型), 113

targeted exploration (有目标的探索), 112

task-hierarchy (任务层次), 300

 partial-order (部分序), 316

taxi (滑行), 123

TD-Gammon, 543

 vs. humans (与人类), 543

TD-Leaf (TD 叶), 547

 exploration (探索), 565

TD-learning (TD 学习)

 in Magic: The Gathering (在 Magic: The Gathering 游戏中), 571

TD-POMDP, 495

temporal credit assignment (时间信用分配), 7

temporal credit assignment problem (时间信用分配问题), 7

temporal difference (时序差分), 511

temporal difference learning (时序差分学习), 29, 543

 and tree search (树搜索), 547

 of value function slope (价值函数斜率), 546

TD(λ), 543

temporal-difference learning (时序差分学习), 226

 GTD2, 228

 Q-learning (Q 学习), 227

 SARSA, 227

 TD-learning (TD 学习), 226

 TDC, 228

terminal state (终止状态), 13

Tetris (俄罗斯方块游戏), 555

 NP-hardness (NP 难), 556

TEXPLORE, 129

Thompson sampling (Thompson 取样), 377
 tile coding (复制编码), 214
 time-to-go (剩余时间), 478
 TOI-Dec-MDP, 494
 topology (拓扑结构), 331
 training regime (培训制度), 565
 database play (数据库播放), 547, 566
 expert training (专家训练), 548
 lesson and practice (课程与实践), 566
 online game server (在线游戏服务器), 548
 self-play (自我对弈), 543, 547, 548, 565
 tutoring (辅导), 566
 transfer learning (迁移学习), 317
 transition function (转换函数), 11, 12
 Markovian (马尔可夫), 11
 tree search (树搜索)
 $\alpha\beta$ -pruning ($\alpha\beta$ 剪枝), 551
 principal leaf (主叶), 547
 TD-Leaf (TD 叶), 547
 TreeStrap, 548
 tutoring (辅导), 566
 TWEANN, 331

U

UCT, 117, 552, 见 Monte-Carlo tree search
 exploration (探索), 565
 narrow path to victory (通往胜利的狭窄道路),
 554
 shallow traps (浅的陷阱), 555

underlying MDP (底层 MDP), 487
 underlying POMDP (底层 POMDP), 487
 Utile Coordination (有效的协调), 462

V

value approximation (价值逼近), 211, 223
 off-policy (离线策略), 223
 offline (离线的), 223
 on-policy (在线策略), 223
 online (在线的), 223
 value function (价值函数)
 as probability of victory (作为胜利的概率), 545
 value function (价值函数), 15
 vs. preference function (与偏好函数), 556, 557
 value function approximation (价值函数逼近)
 weak performance in Tetris (俄罗斯方块的弱性能), 556
 value function decomposition (价值函数分解), 303
 value iteration (值迭代), 23
 value of perfect information (完美信息的价值), 128,
 362, 377
 Vexbot, 568

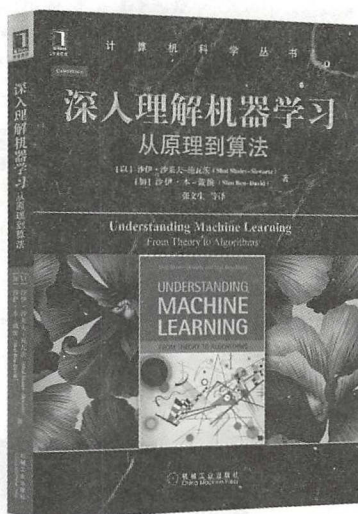
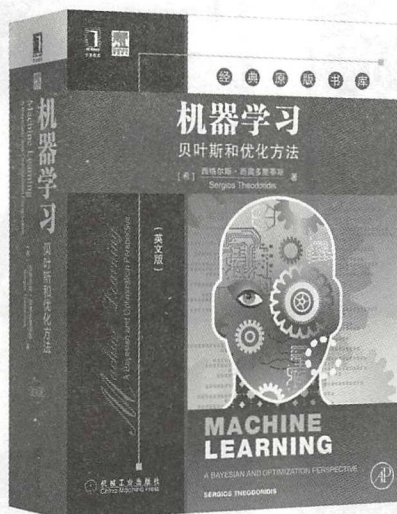
W

Warcraft (战斗机), 558

X

XCS, 337

推荐阅读



机器学习：贝叶斯和优化方法（英文版）

作者：[希] 西格尔斯·西奥多里斯等 ISBN：978-7-111-56526-0 定价：269.00元

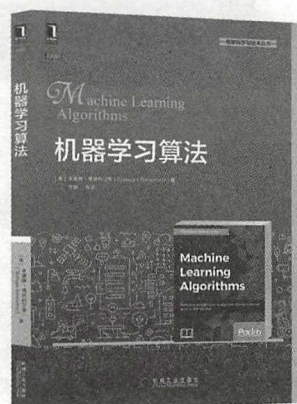
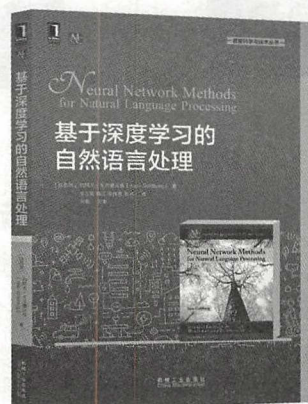
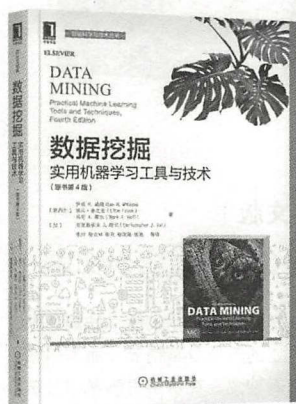
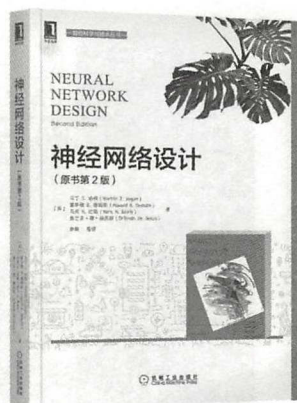
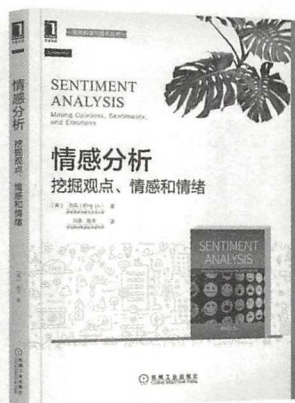
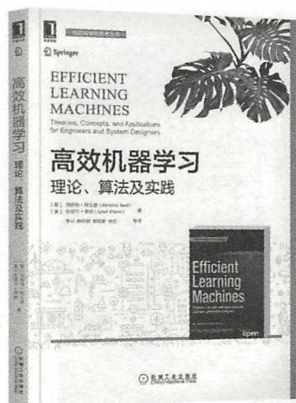
本书对所有主要的机器学习方法和最新研究趋势进行了深入探索，既涵盖基于优化技术的概率和确定性方法，也包含基于层次化概率模型的贝叶斯推断方法。这些背景各异、用途广泛的方法盘根错节，而本书站在全景视角将其一一打通，形成了明晰的机器学习知识体系。书中各章内容相对独立，在讲解机器学习方法时专注于数学理论背后的物理推理，给出数学建模和算法实现，并辅以应用实例和习题，适合该领域的科研人员 and 工程师阅读，也适合学习模式识别、统计/自适应信号处理和深度学习等课程的学生参考。

深入理解机器学习：从原理到算法

作者：[以] 沙伊·沙莱夫-施瓦茨 [加] 沙伊·本-戴维 ISBN：978-7-111-54302-2 定价：79.00元

机器学习是计算机科学中发展最快的领域之一，实际应用广泛。这本教材的目标是从理论角度提供机器学习的入门知识和相关算法范式。本书全面地介绍了机器学习背后的基本思想和理论依据，以及将这些理论转化为实际算法的数学推导。在介绍了机器学习的基本内容后，本书还覆盖了此前的教材中一系列从未涉及过的内容。其中包括对学习的计算复杂度、凸性和稳定性的概念的讨论，以及重要的算法范式的介绍（包括随机梯度下降、神经网络以及结构化输出学习）。同时，本书引入了最新的理论概念，包括PAC-贝叶斯方法和压缩界。本书为高等院校本科高年级和研究生入门阶段而设计，不仅计算机、电子工程、数学统计专业学生能轻松理解机器学习的基础知识和算法，其他专业的读者也能读懂。

智能科学与技术丛书



高效机器学习：理论、算法及实践

作者: Rahul Khanna Mariette Awad ISBN: 978-7-111-56716-5 定价: 69.00元

情感分析：挖掘观点、情感和情绪

作者: Bing Liu ISBN: 978-7-111-57498-9 定价: 99.00元

神经网络设计（原书第2版）

作者: Martin T. Hagan 等 ISBN: 978-7-111-58674-6 定价: 99.00元

数据挖掘：实用机器学习工具与技术（原书第4版）

作者: Ian Witten 等 ISBN: 978-7-111-58916-7 定价: 99.00元

基于深度学习的自然语言处理

作者: Yoav Goldberg ISBN: 978-7-111-59373-7 定价: 99.00元

机器学习算法

作者: Giuseppe Bonaccorso ISBN: 978-7-111-59513-7 定价: 69.00元



Reinforcement Learning State-of-the-Art

强化学习既包括不确定性环境中相关事物的适应性行为，又包括智能学习器在控制、优化和自适应行为等具有挑战性的问题中寻找最优行为的计算方法。其相关领域在过去几十年中取得了巨大的进步。

本书由 17 位不同的领域的专家对强化学习进行了深入而完整的描述，分为六个部分，涉及基础理论到高效解决方案框架的各方面内容，反映了强化学习的主要子领域的研究进展，为读者提供强化学习完整的学习路线，有助于发现新的研究问题和方向。

本书内容

- 强化学习领域经典的方法及高效的解决方案框架，包括 TD 学习、Q 学习、批处理强化学习、策略迭代的最小二乘法、模型的运用以及知识迁移，还分析了更优探索方法的理论优势以获取更好的经验。
- 强化学习中各种表示方式的不同用途，包括基于向量的表示、使用一阶逻辑的表示、有效地运用分层表示以及使用进化算法时用到的无偏表示。
- 强化学习相关的概率框架和算法，包括贝叶斯强化学习框架、部分可观察的马尔可夫决策过程、可预测的状态表示、多学习器的扩展与博弈论的概念、去中心化的部分可观察的马尔可夫决策过程框架及规划算法。
- 强化学习相关领域知识，包括强化学习与人类学习的关系，强化学习在游戏领域的应用以及强化学习在机器人控制中的应用等。

作者介绍

马可·威宁 (Marco Wiering) 荷兰格罗宁根大学人工智能和认知工程系副教授，他发表过各种强化学习主题的文章，研究领域包括强化学习、机器学习、深度学习、计算机视觉等。他主要讲授机器学习、深度学习方面的课程。主持过很多机器学习、人工智能方面的项目。

马丁·范·奥特罗 (Martijn van Otterlo) 是荷兰奈梅亨大学认知人工智能小组的一员。主要研究领域是强化学习在环境中的知识表示。



投稿热线: (010) 88379604
客服热线: (010) 88378991 88361066
购书热线: (010) 68326294 88379649 68995259

华章网站: www.hzbook.com
网上购书: www.china-pub.com
数字阅读: www.hzmedia.com.cn

上架指导: 计算机/机器学习

ISBN 978-7-111-60022-0



9 787111 600220

定价: 119.00元